Mathematical Analysis of Algorithms

Calvin Higgins

Department of Computer Science and Statistics
University of Rhode Island

September 13, 2025

What is algorithm analysis?

Why do we analyze algorithms?

The Big Picture

Algorithm analysis is the prediction and comparison of algorithm performance.

Algorithm analysis lets us choose or design the best (or good enough) algorithm for a given problem.

How do we mathematically analyze algorithms?

Give a step-by-step procedure.

How to Analyze an Algorithm:

1. Define a reasonable **model of computation** (**cost model**).
    1. What are the **basic operations**?
    2. How much does each basic operation cost?
2. Model the algorithm's cost with a function $T(n)$.
    1. How many basic operations are performed for an input of size n?
3. Simplify $T(n)$.
4. Classify $T(n)$'s growth rate.
    1. How quickly does $T(n)$ grow?
5. Interpret $T(n)$'s growth rate.
    1. How suitable is the algorithm for my problem?

Defining a Model of Computation

```
int foo(int n) {
    int A = new int[n];

    A[0] = 1;
    for (int i = 1; i < n; i++)
        A[i] = A[i - 1] * (i + 1);

    int sum = 0;
    for (int i = 0; i < n; i++)
        for (int j = 0; j < i; j++)
            sum += A[j];

    delete[] A;
    return sum;
}
```

List as many basic operations as you can think of!

Defining a Model of Computation

```
int foo(int n) {
    int* A = new int[n];

    A[0] = 1;
    for (int i = 1; i < n; i++)
        A[i] = A[i - 1] * (i + 1);

    int sum = 0;
    for (int i = 0; i < n; i++)
        for (int j = 0; j < i; j++)
            sum += A[j];

    delete[] A;
    return sum;
}
```

Basic Operations:

1. Additions
2. Multiplications
3. Comparisons
4. Branches
5. Local variables
6. Memory allocations
7. Allocated memory
8. Loads
9. Stores
10. Assignments
11. ...

Which basic operations are most reasonable? Why?

Modeling the Algorithm's Cost

```
int foo(int n) {
    int* A = new int[n];

    A[0] = 1;
    for (int i = 1; i < n; i++)          // Additions?
        A[i] = A[i - 1] * (i + 1);      // Additions?

    int sum = 0;
    for (int i = 0; i < n; i++)          // Additions?
        for (int j = 0; j < i; j++)     // Additions?
            sum += A[j];                 // Additions?

    delete[] A;
    return sum;
}
```

How many additions? $T(n) = ?$

Modeling the Algorithm's Cost

```
int foo(int n) {
    int* A = new int[n];

    A[0] = 1;
    for (int i = 1; i < n; i++)          // ∑_{i=1}^{n-1} 1 additions
        A[i] = A[i - 1] * (i + 1);       // ∑_{i=1}^{n-1} 2 additions

    int sum = 0;
    for (int i = 0; i < n; i++)          // ∑_{i=0}^{n-1} 1 additions
        for (int j = 0; j < i; j++)      // ∑_{i=0}^{n-1} ∑_{j=0}^{i} 1 additions
            sum += A[j];                 // ∑_{i=0}^{n-1} ∑_{j=0}^{i-1} 1 additions

    delete[] A;
    return sum;
}
```

$$T(n) = \sum_{i=1}^{n-1} 1 + \sum_{i=1}^{n-1} 2 + \sum_{i=0}^{n-1} 1 + \sum_{i=0}^{n-1} \sum_{j=0}^{i} 1 + \sum_{i=0}^{n-1} \sum_{j=0}^{i-1} 1$$

Modeling the Algorithm's Cost

```
int foo(int n) {
    int* A = new int[n];

    A[0] = 1;
    for (int i = 1; i < n; i++)
        A[i] = A[i - 1] * (i + 1); // Multiplications?

    int sum = 0;
    for (int i = 0; i < n; i++)
        for (int j = 0; j < i; j++)
            sum += A[j];

    delete[] A;
    return sum;
}
```

How many multiplications? $T(n) = ?$

Modeling the Algorithm's Cost

```
int foo(int n) {
    int* A = new int[n];

    A[0] = 1;
    for (int i = 1; i < n; i++)
        A[i] = A[i - 1] * (i + 1); // ∑_{i=1}^{n-1} 1 multiplications

    int sum = 0;
    for (int i = 0; i < n; i++)
        for (int j = 0; j < i; j++)
            sum += A[j];

    delete[] A;
    return sum;
}
```

Multiplications: $T(n) = \sum_{i=1}^{n-1} 1$

Modeling the Algorithm's Cost

```
int foo(int n) {
    int* A = new int[n];

    A[0] = 1;
    for (int i = 1; i < n; i++)       // Comparisons?
        A[i] = A[i - 1] * (i + 1);

    int sum = 0;
    for (int i = 0; i < n; i++)       // Comparisons?
        for (int j = 0; j < i; j++) // Comparisons?
            sum += A[j];

    delete[] A;
    return sum;
}
```

How many comparisons? $T(n) = ?$

Modeling the Algorithm's Cost

```
int foo(int n) {
    int* A = new int[n];

    A[0] = 1;
    for (int i = 1; i < n; i++)          // ∑_{i=1}^{n} 1 comparisons
        A[i] = A[i - 1] * (i + 1);

    int sum = 0;
    for (int i = 0; i < n; i++)          // ∑_{i=0}^{n} 1 comparisons
        for (int j = 0; j < i; j++)  // ∑_{i=0}^{n-1} ∑_{j=0}^{i} 1 comparisons
            sum += A[j];

    delete[] A;
    return sum;
}
```

Comparisons: $T(n) = \sum_{i=1}^{n} 1 + \sum_{i=0}^{n} 1 + \sum_{i=0}^{n-1} \sum_{j=0}^{i} 1$

Modeling the Algorithm's Cost

```
int foo(int n) {
    int* A = new int[n];

    A[0] = 1;                          // Memory accesses?
    for (int i = 1; i < n; i++)
        A[i] = A[i - 1] * (i + 1);     // Memory accesses?

    int sum = 0;
    for (int i = 0; i < n; i++)
        for (int j = 0; j < i; j++)
            sum += A[j];               // Memory accesses?

    delete[] A;
    return sum;
}
```

How many memory accesses (indexing)? $T(n) = ?$

Modeling the Algorithm's Cost

```
int foo(int n) {
    int* A = new int[n];

    A[0] = 1;                          // 1 memory accesses
    for (int i = 1; i < n; i++)
        A[i] = A[i - 1] * (i + 1);     // ∑_{i=1}^{n-1} 2 memory accesses

    int sum = 0;
    for (int i = 0; i < n; i++)
        for (int j = 0; j < i; j++)
            sum += A[j];               // ∑_{i=0}^{n-1} ∑_{j=0}^{i-1} 1 memory access

    delete[] A;
    return sum;
}
```

Memory Accesses: $T(n) = 1 + \sum_{i=1}^{n-1} 2 + \sum_{i=0}^{n-1} \sum_{j=0}^{i-1} 1$

Modeling the Algorithm's Cost

```
int foo(int n) {
    int* A = new int[n];                        // Assignments?

    A[0] = 1;                                   // Assignments?
    for (int i = 1; i < n; i++)                 // Assignments?
        A[i] = A[i - 1] * (i + 1);              // Assignments?

    int sum = 0;                                // Assignments?
    for (int i = 0; i < n; i++)                 // Assignments?
        for (int j = 0; j < i; j++)             // Assignments?
            sum += A[j];                        // Assignments?

    delete[] A;
    return sum;
}
```

How many assignments? $T(n) = ?$

Modeling the Algorithm's Cost

```
int foo(int n) {
    int* A = new int[n];                     // 1 assignments

    A[0] = 1;                                // 1 assignments
    for (int i = 1; i < n; i++)              // Σ_{i=1}^{n-1} 1 assignments
        A[i] = A[i - 1] * (i + 1);           // Σ_{i=1}^{n-1} 1 assignments

    int sum = 0;                             // 1 assignments
    for (int i = 0; i < n; i++)              // Σ_{i=0}^{n-1} 1 assignments
        for (int j = 0; j < i; j++)          // Σ_{i=0}^{n-1} Σ_{j=0}^{i} 1 assignments
            sum += A[j];                     // Σ_{i=0}^{n-1} Σ_{j=0}^{i-1} 1 assignments

    delete[] A;
    return sum;
}
```

$$T(n) = 3 + \sum_{i=1}^{n} 1 + \sum_{i=1}^{n-1} 1 + \sum_{i=0}^{n} 1 + \sum_{i=0}^{n-1} \sum_{j=0}^{i} 1 + \sum_{i=0}^{n-1} \sum_{j=0}^{i-1} 1$$

Simplifying the Cost Function

Addition Cost Function

$$T(n) = \sum_{i=1}^{n-1} 1 + \sum_{i=1}^{n-1} 2 + \sum_{i=0}^{n-1} 1 + \sum_{i=0}^{n-1} \sum_{j=0}^{i} 1 + \sum_{i=0}^{n-1} \sum_{j=0}^{i-1} 1$$

Simplification:

Apply (1) with $m = n - 1$:

$$\sum_{i=1}^{n-1} 1 = n - 1$$

Identities:

1. $\sum_{i=1}^{m} 1 = m$
2. $\sum_{i=1}^{m} i = \frac{m(m+1)}{2}$
3. $\sum_{i=a}^{b} cf(i) = c \sum_{i=a}^{b} f(i)$ where c is a **constant** and f is a function
4. $\sum_{i=a}^{b} (f(i) + g(i)) = \sum_{i=a}^{b} f(i) + \sum_{i=a}^{b} g(i)$ where f and g are functions

Simplifying the Cost Function

Addition Cost Function

$$T(n) = (n-1) + \sum_{i=1}^{n-1} 2 + \sum_{i=0}^{n-1} 1 + \sum_{i=0}^{n-1} \sum_{j=0}^{i} 1 + \sum_{i=0}^{n-1} \sum_{j=0}^{i-1} 1$$

Simplification:

$$\sum_{i=1}^{n-1} 2 = ?$$

What identity should we use?

Identities:

1. $\sum_{i=1}^{m} 1 = m$
2. $\sum_{i=1}^{m} i = \frac{m(m+1)}{2}$
3. $\sum_{i=a}^{b} cf(i) = c \sum_{i=a}^{b} f(i)$ where c is a **constant** and f is a function
4. $\sum_{i=a}^{b} (f(i) + g(i)) = \sum_{i=a}^{b} f(i) + \sum_{i=a}^{b} g(i)$ where f and g are functions

Simplifying the Cost Function

Addition Cost Function

$$T(n) = (n-1) + \sum_{i=1}^{n-1} 2 + \sum_{i=0}^{n-1} 1 + \sum_{i=0}^{n-1} \sum_{j=0}^{i} 1 + \sum_{i=0}^{n-1} \sum_{j=0}^{i-1} 1$$

Simplification:

Apply (3) with $a = 1$, $b = n-1$, $c = 2$ and $f(i) = 1$

$$\sum_{i=1}^{n-1} 2 = \sum_{i=1}^{n-1} 2 \cdot 1 = 2 \sum_{i=1}^{n-1} 1$$

What identity should we use?

Identities:

1. $\sum_{i=1}^{m} 1 = m$
2. $\sum_{i=1}^{m} i = \frac{m(m+1)}{2}$
3. $\sum_{i=a}^{b} cf(i) = c \sum_{i=a}^{b} f(i)$ where c is a **constant** and f is a function
4. $\sum_{i=a}^{b} (f(i) + g(i)) = \sum_{i=a}^{b} f(i) + \sum_{i=a}^{b} g(i)$ where f and g are functions

Addition Cost Function

$$T(n) = (n-1) + \sum_{i=1}^{n-1} 2 + \sum_{i=0}^{n-1} 1 + \sum_{i=0}^{n-1}\sum_{j=0}^{i} 1 + \sum_{i=0}^{n-1}\sum_{j=0}^{i-1} 1$$

Simplification:

Apply (3) with $a = 1$, $b = n-1$, $c = 2$ and $f(i) = 1$

$$\sum_{i=1}^{n-1} 2 = \sum_{i=1}^{n-1} 2 \cdot 1 = 2\sum_{i=1}^{n-1} 1$$

Apply (1) with $m = n-1$

$$2\sum_{i=1}^{n-1} 1 = 2(n-1)$$

Identities:

1. $\sum_{i=1}^{m} 1 = m$
2. $\sum_{i=1}^{m} i = \frac{m(m+1)}{2}$
3. $\sum_{i=a}^{b} cf(i) = c\sum_{i=a}^{b} f(i)$ where c is a **constant** and f is a function
4. $\sum_{i=a}^{b}(f(i)+g(i)) = \sum_{i=a}^{b} f(i) + \sum_{i=a}^{b} g(i)$ where f and g are functions

Simplifying the Cost Function

Addition Cost Function

$$T(n) = (n-1) + 2(n-1) + \sum_{i=0}^{n-1} 1 + \sum_{i=0}^{n-1} \sum_{j=0}^{i} 1 + \sum_{i=0}^{n-1} \sum_{j=0}^{i-1} 1$$

Simplification:

$$\sum_{i=0}^{n-1} 1 = ?$$

What identity should we use?

Identities:

1. $\sum_{i=1}^{m} 1 = m$
2. $\sum_{i=1}^{m} i = \frac{m(m+1)}{2}$
3. $\sum_{i=a}^{b} cf(i) = c \sum_{i=a}^{b} f(i)$ where c is a **constant** and f is a function
4. $\sum_{i=a}^{b} (f(i) + g(i)) = \sum_{i=a}^{b} f(i) + \sum_{i=a}^{b} g(i)$ where f and g are functions

Simplifying the Cost Function

Addition Cost Function

$$T(n) = (n-1) + 2(n-1) + \sum_{i=0}^{n-1} 1 + \sum_{i=0}^{n-1} \sum_{j=0}^{i} 1 + \sum_{i=0}^{n-1} \sum_{j=0}^{i-1} 1$$

Simplification:

Apply (1) with $m = n - 1$

$$\sum_{i=0}^{n-1} 1 = 1 + \sum_{i=1}^{n-1} 1 = (n-1) + 1 = n$$

Identities:

1. $\sum_{i=1}^{m} 1 = m$
2. $\sum_{i=1}^{m} i = \frac{m(m+1)}{2}$
3. $\sum_{i=a}^{b} cf(i) = c \sum_{i=a}^{b} f(i)$ where c is a **constant** and f is a function
4. $\sum_{i=a}^{b} (f(i) + g(i)) = \sum_{i=a}^{b} f(i) + \sum_{i=a}^{b} g(i)$ where f and g are functions

Addition Cost Function

$$T(n) = (n-1) + 2(n-1) + n + \sum_{i=0}^{n-1}\sum_{j=0}^{i} 1 + \sum_{i=0}^{n-1}\sum_{j=0}^{i-1} 1$$

Simplification:

$$\sum_{i=0}^{n-1}\sum_{j=0}^{i} 1 = ?$$

What identity should we use?

Identities:

1. $\sum_{i=1}^{m} 1 = m$
2. $\sum_{i=1}^{m} i = \frac{m(m+1)}{2}$
3. $\sum_{i=a}^{b} cf(i) = c\sum_{i=a}^{b} f(i)$ where c is a **constant** and f is a function
4. $\sum_{i=a}^{b} (f(i) + g(i)) = \sum_{i=a}^{b} f(i) + \sum_{i=a}^{b} g(i)$ where f and g are functions

Simplifying the Cost Function

Addition Cost Function

$$T(n) = (n-1) + 2(n-1) + n + \sum_{i=0}^{n-1} \sum_{j=0}^{i} 1 + \sum_{i=0}^{n-1} \sum_{j=0}^{i-1} 1$$

Simplification:

Apply (1) with $m = i$

$$\sum_{i=0}^{n-1} \sum_{j=0}^{i} 1 = \sum_{i=0}^{n-1} \left(1 + \sum_{j=1}^{i} 1 \right) = \sum_{i=0}^{n-1} (i+1)$$

What identity should we use?

Identities:

1. $\sum_{i=1}^{m} 1 = m$
2. $\sum_{i=1}^{m} i = \frac{m(m+1)}{2}$
3. $\sum_{i=a}^{b} cf(i) = c \sum_{i=a}^{b} f(i)$ where c is a **constant** and f is a function
4. $\sum_{i=a}^{b} (f(i) + g(i)) = \sum_{i=a}^{b} f(i) + \sum_{i=a}^{b} g(i)$ where f and g are functions

Simplifying the Cost Function

Addition Cost Function

$$T(n) = (n-1) + 2(n-1) + n + \sum_{i=0}^{n-1} \sum_{j=0}^{i} 1 + \sum_{i=0}^{n-1} \sum_{j=0}^{i-1} 1$$

Simplification:

Apply (4) with $a = 0$, $b = n-1$, $f(i) = i$ and $g(i) = 1$

$$\sum_{i=0}^{n-1} (i+1) = \sum_{i=0}^{n-1} i + \sum_{i=0}^{n-1} 1$$

What identities should we use?

Identities:

1. $\sum_{i=1}^{m} 1 = m$
2. $\sum_{i=1}^{m} i = \frac{m(m+1)}{2}$
3. $\sum_{i=a}^{b} cf(i) = c \sum_{i=a}^{b} f(i)$ where c is a **constant** and f is a function
4. $\sum_{i=a}^{b} (f(i) + g(i)) = \sum_{i=a}^{b} f(i) + \sum_{i=a}^{b} g(i)$ where f and g are functions

Simplifying the Cost Function

Addition Cost Function

$$T(n) = (n-1) + 2(n-1) + n + \sum_{i=0}^{n-1} \sum_{j=0}^{i} 1 + \sum_{i=0}^{n-1} \sum_{j=0}^{i-1} 1$$

Simplification:

Apply (1) with $m = n - 1$

$$\sum_{i=0}^{n-1} 1 = 1 + \sum_{i=1}^{n-1} 1$$
$$= 1 + (n-1) = n$$

so

$$\sum_{i=0}^{n-1} i + \sum_{i=0}^{n-1} 1 = n + \sum_{i=0}^{n-1} i$$

Identities:

1. $\sum_{i=1}^{m} 1 = m$
2. $\sum_{i=1}^{m} i = \frac{m(m+1)}{2}$
3. $\sum_{i=a}^{b} cf(i) = c \sum_{i=a}^{b} f(i)$ where c is a **constant** and f is a function
4. $\sum_{i=a}^{b} (f(i) + g(i)) = \sum_{i=a}^{b} f(i) + \sum_{i=a}^{b} g(i)$ where f and g are functions

Simplifying the Cost Function

Addition Cost Function

$$T(n) = (n-1) + 2(n-1) + n + \sum_{i=0}^{n-1}\sum_{j=0}^{i} 1 + \sum_{i=0}^{n-1}\sum_{j=0}^{i-1} 1$$

Simplification:

Apply (2) with $m = n - 1$

$$\sum_{i=0}^{n-1} i = 0 + \sum_{i=1}^{n-1} i$$

$$= \frac{(n-1)((n-1)+1)}{2}$$

so

$$n + \sum_{i=0}^{n-1} i = n + \frac{n(n-1)}{2}$$

Identities:

1. $\sum_{i=1}^{m} 1 = m$
2. $\sum_{i=1}^{m} i = \frac{m(m+1)}{2}$
3. $\sum_{i=a}^{b} cf(i) = c\sum_{i=a}^{b} f(i)$ where c is a **constant** and f is a function
4. $\sum_{i=a}^{b}(f(i)+g(i)) = \sum_{i=a}^{b} f(i) + \sum_{i=a}^{b} g(i)$ where f and g are functions

Simplifying the Cost Function

Addition Cost Function

$$T(n) = (n-1) + 2(n-1) + n + \left(n + \frac{n(n-1)}{2}\right) + \sum_{i=0}^{n-1}\sum_{j=0}^{i-1} 1$$

Simplification:

$$\sum_{i=0}^{n-1}\sum_{j=0}^{i-1} 1 = ?$$

What is the simplified form?

Identities:

1. $\sum_{i=1}^{m} 1 = m$
2. $\sum_{i=1}^{m} i = \frac{m(m+1)}{2}$
3. $\sum_{i=a}^{b} cf(i) = c\sum_{i=a}^{b} f(i)$ where c is a **constant** and f is a function
4. $\sum_{i=a}^{b} (f(i) + g(i)) = \sum_{i=a}^{b} f(i) + \sum_{i=a}^{b} g(i)$ where f and g are functions

Simplifying the Cost Function

Addition Cost Function

$$T(n) = (n-1) + 2(n-1) + n + \left(n + \frac{n(n-1)}{2}\right) + \sum_{i=0}^{n-1}\sum_{j=0}^{i-1} 1$$

Simplification:

Apply (1) then (2)

$$\sum_{i=0}^{n-1}\sum_{j=0}^{i-1} 1 = \sum_{i=0}^{n-1} i = \frac{n(n-1)}{2}$$

Identities:

1. $\sum_{i=1}^{m} 1 = m$
2. $\sum_{i=1}^{m} i = \frac{m(m+1)}{2}$
3. $\sum_{i=a}^{b} cf(i) = c\sum_{i=a}^{b} f(i)$ where c is a **constant** and f is a function
4. $\sum_{i=a}^{b} (f(i) + g(i)) = \sum_{i=a}^{b} f(i) + \sum_{i=a}^{b} g(i)$ where f and g are functions

Simplifying the Cost Function

Simplification:
We have that

$$\begin{aligned} T(n) &= (n-1) + 2(n-1) + n + \left(n + \frac{n(n-1)}{2}\right) + \frac{n(n-1)}{2} \\ &= n - 1 + 2n - 2 + n + n + n(n-1) \\ &= n^2 + 4n - 3 \end{aligned}$$

so the final answer is

$$T(n) = n^2 + 4n - 3$$

Simplifying the Cost Function

Multiplication Cost Function

$T(n) = \sum_{i=1}^{n-1} 1$

Comparisons Cost Function

$T(n) = \sum_{i=1}^{n} 1 + \sum_{i=0}^{n} 1 + \sum_{i=0}^{n-1} \sum_{j=0}^{i} 1$

Memory Accesses Cost Function

$T(n) = 1 + \sum_{i=1}^{n-1} 2 + \sum_{i=0}^{n-1} \sum_{j=0}^{i-1} 1$

Assignments Cost Function

$T(n) = 3 + \sum_{i=1}^{n-1} 1 + \sum_{i=1}^{n-1} 1 + \sum_{i=0}^{n-1} 1 + \sum_{i=0}^{n-1} \sum_{j=0}^{i} 1 + \sum_{i=0}^{n-1} \sum_{j=0}^{i-1} 1$

Try to simplify the other cost functions!

Simplifying the Cost Function

Multiplication Cost Function

$$\begin{aligned} T(n) &= \sum_{i=1}^{n-1} 1 \\ &= n - 1 \end{aligned}$$

Simplifying the Cost Function

Comparisons Cost Function

$$
\begin{aligned}
T(n) &= \sum_{i=1}^{n} 1 + \sum_{i=0}^{n} 1 + \sum_{i=0}^{n-1} \sum_{j=0}^{i} 1 \\
&= n + (n+1) + \frac{n(n+1)}{2} \\
&= \frac{1}{2}n^2 + \frac{5}{2}n + 1
\end{aligned}
$$

Simplifying the Cost Function

Memory Accesses Cost Function

$$
\begin{aligned}
T(n) &= 1 + \sum_{i=1}^{n-1} 2 + \sum_{i=0}^{n-1} \sum_{j=0}^{i-1} 1 \\
&= 1 + 2(n-1) + \frac{n(n-1)}{2} \\
&= \frac{1}{2} n^2 + \frac{3}{2} n - 1
\end{aligned}
$$

Simplifying the Cost Function

Assignments Cost Function

$$
\begin{aligned}
T(n) &= 3 + \sum_{i=1}^{n-1} 1 + \sum_{i=1}^{n-1} 1 + \sum_{i=0}^{n-1} 1 + \sum_{i=0}^{n-1} \sum_{j=0}^{i} 1 + \sum_{i=0}^{n-1} \sum_{j=0}^{i-1} 1 \\
&= 3 + (n-1) + (n-1) + n + \frac{n(n+1)}{2} + \frac{n(n-1)}{2} \\
&= n^2 + 3n + 1
\end{aligned}
$$

Simplifying the Cost Function

Pitfall: Cannot represent loops with non-one increments using standard summations!

```
int baz(int n) {
    int total = 0;
    for (int i = 0; i < n; i += 2)
        total += i * i;
    return total;
}
```

Solution: Compute number of operations for small values of n, then guess and check the formula OR learn Knuth's summation notation (advanced)!

Extra Practice

```
int bar(int n) {
    int* A = new int[n];

    for (int i = 0; i < n; i++)
        A[i] = i + 1;

    int result = 0;
    for (int k = 0; k < n * n; k++)
        for (int j = 0; j <= k; j++)
            for (int i = j; i < n; i++)
                result += A[i];

    for (int t = 0; t < 7; t++)
        result += A[0];

    delete[] A;
    return result;
}
```

Choose a model of computation and analyze this algorithm!