Fast integer and matrix multiplication algorithms

Aadish Verma

July 10, 2026

Goals

1. Intuition for theoretical computer science

Goals

1. Intuition for theoretical computer science

2. Appreciation for depth of algorithms

Goals

1. Intuition for theoretical computer science

2. Appreciation for depth of algorithms

3. Maybe an algorithm or two!

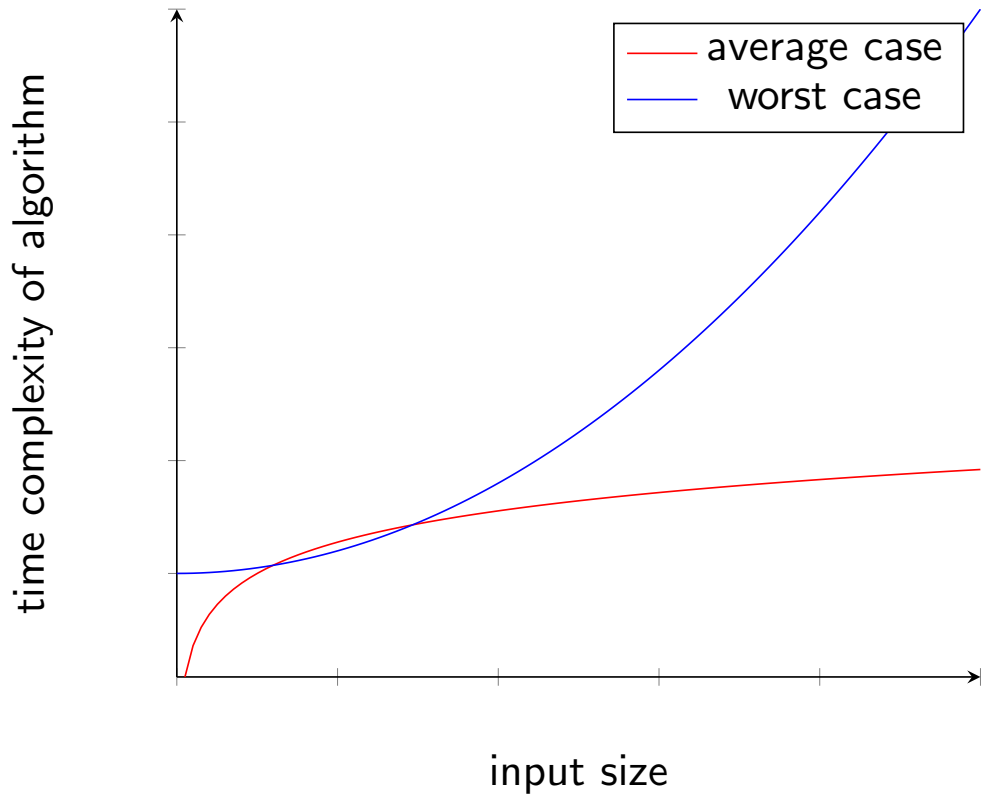
Thinking in theoretical CS

For our purposes, theoretical computer science is the field of finding and analyzing algorithms.

Thinking in theoretical CS

When thinking about the speed of algorithms, we have two major "axioms":

1. We always think in terms of the worst case.

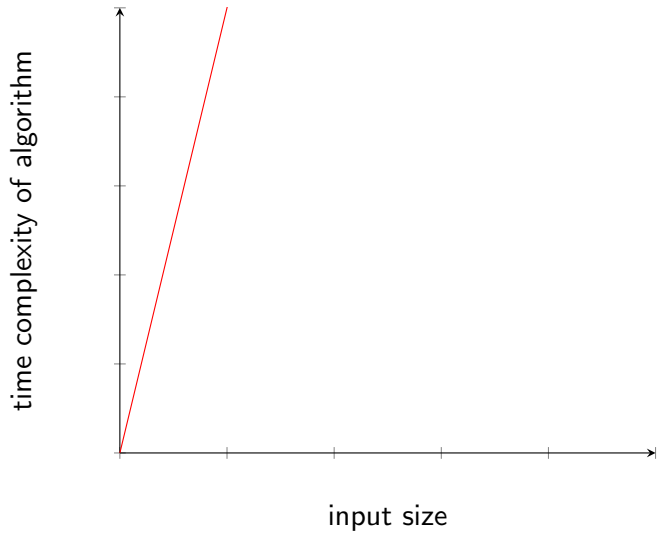


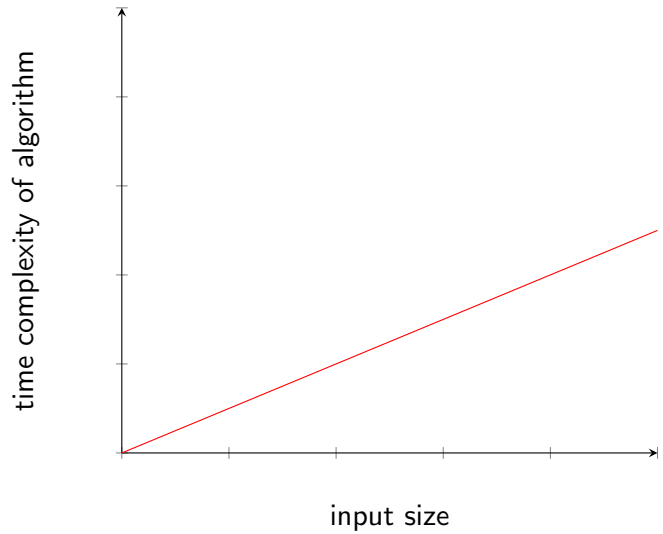
Thinking in theoretical CS

When thinking about the speed of algorithms, we have two major "axioms":

2. We only care about asymptotic behavior.

For example, the following two complexities are equivalent for our purposes:



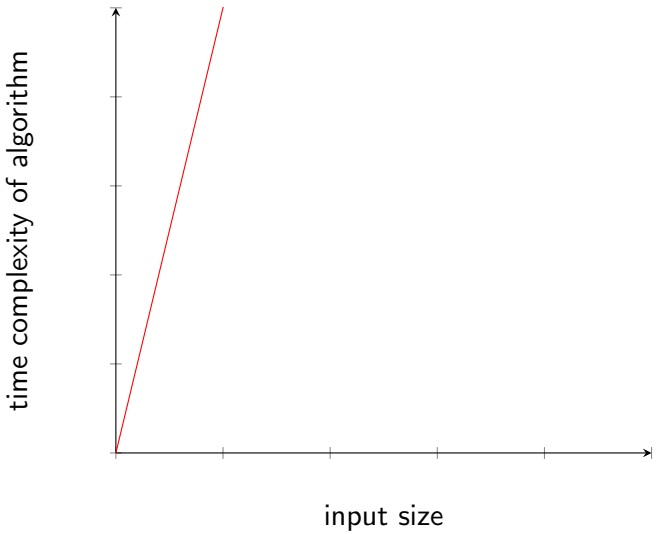


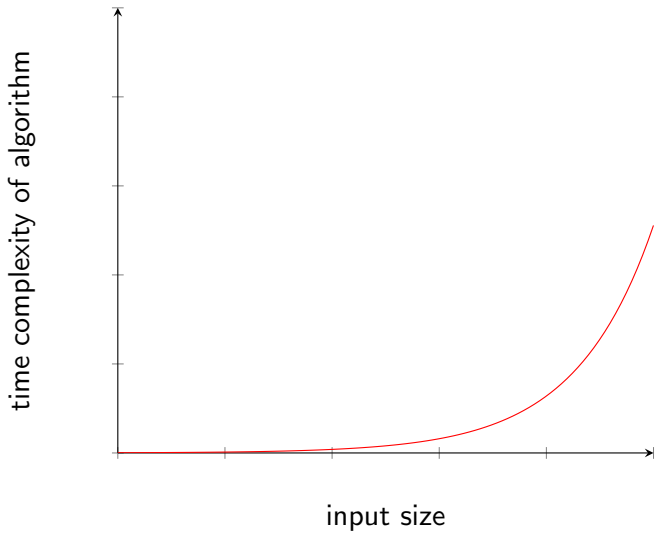
Thinking in theoretical CS

When thinking about the speed of algorithms, we have two major "axioms":

2. We only care about asymptotic behavior.

And, counterintuitively, we treat the graph on the left as faster than that on the right:





My area

1. Integer multiplication

My area

1. Integer multiplication

2. Matrix multiplication

My area

1. Integer multiplication ($O(n^2)$ naively)

2. Matrix multiplication ($O(n^3)$ naively)

My area

1. Integer multiplication ($O(n^2)$ naively) **(now $O(n \log n)$)**

2. Matrix multiplication ($O(n^3)$ naively) **(now $O(n^{2.371339})$)**

My area

1. Integer multiplication ($O(n^2)$ naively) **(now $O(n \log n)$)**

2. Matrix multiplication ($O(n^3)$ naively) **(now $O(n^{2.371339})$)**

   However, both of these theoretical upper bounds result in *galactic algorithms*.

Example: Toom-Cook

1. Express inputs X and Y as polynomials where each block of digits is a coefficient

$$X = 123, Y = 456$$
$$p(b) = 1b^2 + 2b + 3$$
$$q(b) = 4b^2 + 5b + 6.$$

Note

$$p(10) = 123 = X, q(10) = 456 = Y.$$

Example: Toom-Cook

1. Express inputs X and Y as polynomials where each block of digits is a coefficient

$$X = 123, Y = 456$$
$$p(b) = 1b^2 + 2b + 3$$
$$q(b) = 4b^2 + 5b + 6.$$

Note

$$p(10) = 123 = X, q(10) = 456 = Y.$$

The big idea:

$$r(b) = p(b)q(b)$$
$$r(10) = p(10)q(10) = XY.$$

If we find the coefficients of r using interpolation, we can just plug in 10 to compute the output.

Example: Toom-Cook

2. Find input-output pairs for p & q, then r

$$p(0) = 3, p(1) = 6, p(-1) = 2, p(2) = 11, p(-2) = 3$$

$$q(0) = 6, q(1) = 15, q(-1) = 5, q(2) = 32, q(-2) = 12.$$

Example: Toom-Cook

2. Find input-output pairs for p & q, then r

$$p(0) = 3, p(1) = 6, p(-1) = 2, p(2) = 11, p(-2) = 3$$
$$q(0) = 6, q(1) = 15, q(-1) = 5, q(2) = 32, q(-2) = 12.$$

Then,

$$r(0) = p(0)q(0) = 18, \ldots$$

Example: Toom-Cook

2. Find input-output pairs for p & q, then r

$$p(0) = 3, p(1) = 6, p(-1) = 2, p(2) = 11, p(-2) = 3$$
$$q(0) = 6, q(1) = 15, q(-1) = 5, q(2) = 32, q(-2) = 12.$$

Then,

$$r(0) = p(0)q(0) = 18, \ldots$$

Divide-and-conquer: use this *same algorithm* to compute the products of p and q.

Example: Toom-Cook

3. Use multiplication by inverse matrix to compute coefficients of r

   Define the coefficients

$$r(b) = c_4 b^4 + c_3 b^3 + c_2 b^2 + c_1 b + c_0.$$

   Then:

$$\begin{bmatrix} i_1^4 & i_1^3 & i_1^2 & i_1^1 & i_1^0 \\ i_2^4 & i_2^3 & i_2^2 & i_2^1 & i_2^0 \\ i_3^4 & i_3^3 & i_3^2 & i_3^1 & i_3^0 \\ i_4^4 & i_4^3 & i_4^2 & i_4^1 & i_4^0 \\ i_5^4 & i_5^3 & i_5^2 & i_5^1 & i_5^0 \end{bmatrix} \begin{bmatrix} c_4 \\ c_3 \\ c_2 \\ c_1 \\ c_0 \end{bmatrix} = \begin{bmatrix} r(i_1) \\ r(i_2) \\ r(i_3) \\ r(i_4) \\ r(i_5) \end{bmatrix}.$$

Example: Toom-Cook

3. Use multiplication by inverse matrix to compute coefficients of r

   Define the coefficients

$$r(b) = c_4 b^4 + c_3 b^3 + c_2 b^2 + c_1 b + c_0.$$

   Then:

$$\begin{bmatrix} c_4 \\ c_3 \\ c_2 \\ c_1 \\ c_0 \end{bmatrix} = \begin{bmatrix} r(i_1) \\ r(i_2) \\ r(i_3) \\ r(i_4) \\ r(i_5) \end{bmatrix} \left(\begin{bmatrix} i_1^4 & i_1^3 & i_1^2 & i_1^1 & i_1^0 \\ i_2^4 & i_2^3 & i_2^2 & i_2^1 & i_2^0 \\ i_3^4 & i_3^3 & i_3^2 & i_3^1 & i_3^0 \\ i_4^4 & i_4^3 & i_4^2 & i_4^1 & i_4^0 \\ i_5^4 & i_5^3 & i_5^2 & i_5^1 & i_5^0 \end{bmatrix} \right)^{-1}.$$

Example: Toom-Cook

3. Use multiplication by inverse matrix to compute coefficients of r

$$r(b) = 4b^4 + 13b^3 + 28b^2 + 27b + 18.$$

Example: Toom-Cook

3. Use multiplication by inverse matrix to compute coefficients of r

$$r(b) = 4b^4 + 13b^3 + 28b^2 + 27b + 18.$$

4. Substitute in for b and evaluate r to compute XY

$$\boxed{r(10) = 56088 = 123 \times 456.}$$

Example: Toom-Cook

3. Use multiplication by inverse matrix to compute coefficients of r

$$r(b) = 4b^4 + 13b^3 + 28b^2 + 27b + 18.$$

4. Substitute in for b and evaluate r to compute XY

$$\boxed{r(10) = 56088 = 123 \times 456.}$$

5. Toom-Cook multiplication has an time complexity of

$$O(n^{\log_k(2k-1)})$$

($\approx O(n^{1.465})$ for our example).

Example: Matrix multiplication

1. Multiplication of two $n \times n$ matrices can be represented with a (usually) sparse $n^2 \times n^2 \times n^2$ tensor. Multiplication becomes equivalent to tensor contraction.

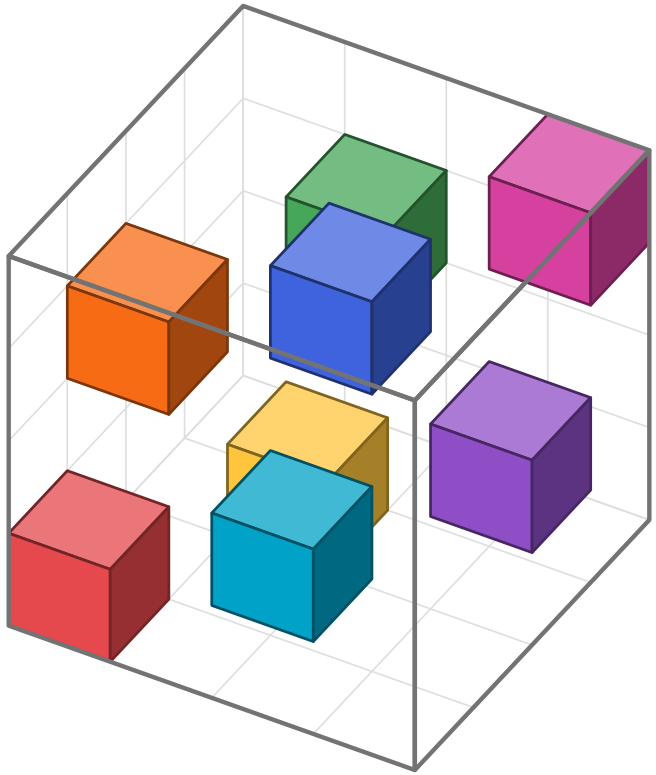


Figure: 2×2 matrix multiplication tensor; each colored cube represents an entry of 1, while all other entries are 0.

Example: Matrix multiplication

1. Multiplication of two $n \times n$ matrices can be represented with a (usually) sparse $n^2 \times n^2 \times n^2$ tensor. Multiplication becomes equivalent to tensor contraction.
2. Decomposition of a matmul tensor into R rank-1 tensors is equivalent to a matrix multiplication algorithm with R bilinear multiplications.

Remark

Why do we only care about bilinear multiplications?

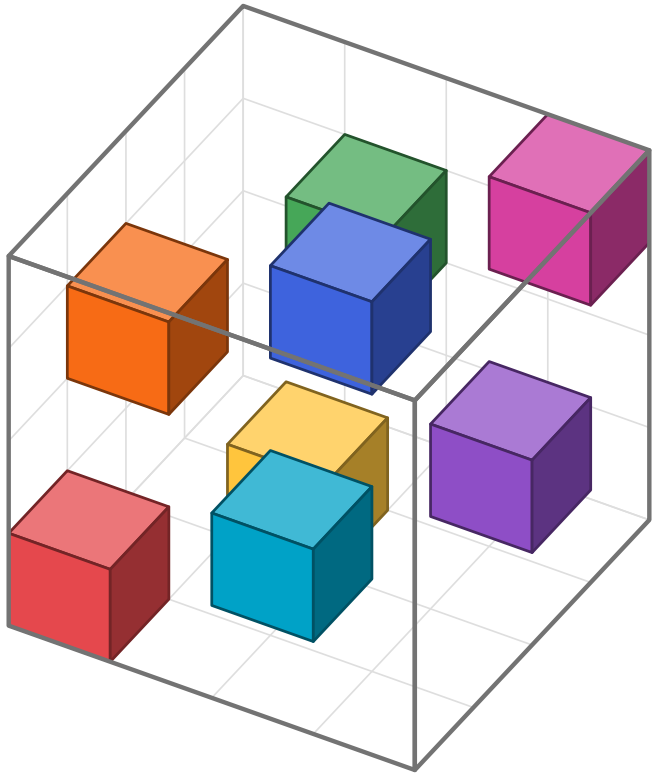


Figure: 2×2 matrix multiplication tensor; each colored cube represents an entry of 1, while all other entries are 0.

Example: Matrix multiplication

3. More efficient decompositions (with lower R) lead to faster matmul algorithms.

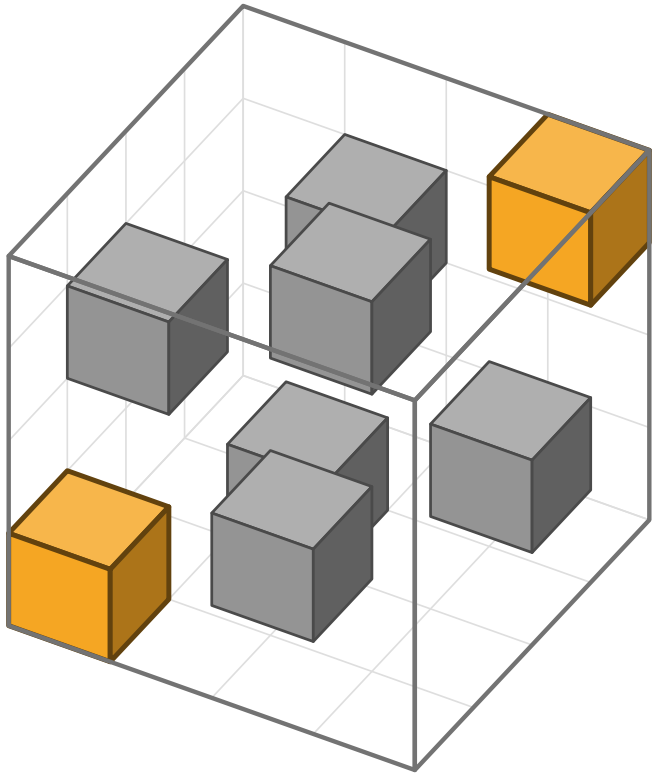


Figure: Strassen's 1969 decomposition of the same 2×2 matrix multiplication tensor, this time into only 7 rank-1 subtensors.

Example: Matrix multiplication

3. More efficient decompositions (with lower R) lead to faster matmul algorithms.
4. Strassen's laser method (not pictured): same general idea, way more complex.

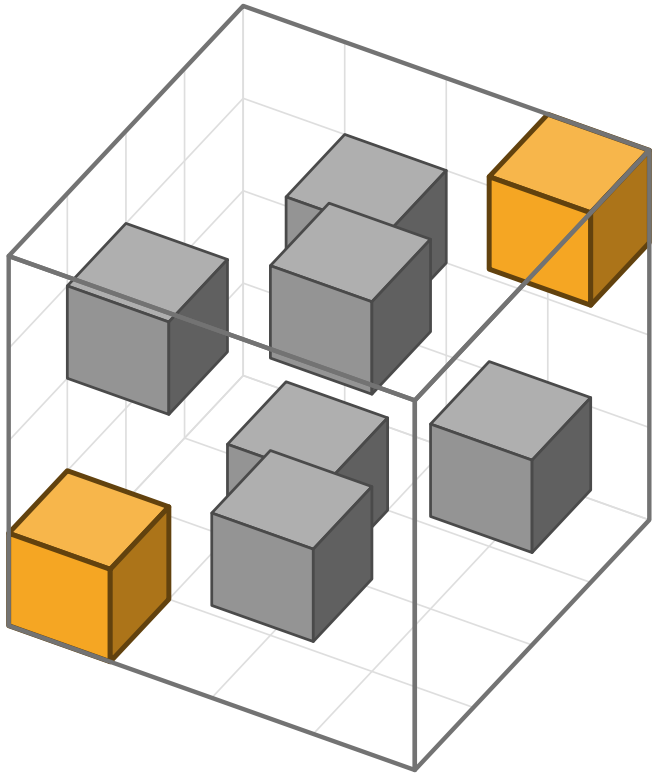


Figure: Strassen's 1969 decomposition of the same 2×2 matrix multiplication tensor, this time into only 7 rank-1 subtensors.

Takeaways

1. These algorithms are for *massive n* (Toom-Cook crossover is at numbers with hundreds of digits)

Takeaways

1. These algorithms are for *massive n* (Toom-Cook crossover is at numbers with hundreds of digits)
2. Algorithms often work with abstract mathematical representations of a problem

Takeaways

1. These algorithms are for *massive n* (Toom-Cook crossover is at numbers with hundreds of digits)
2. Algorithms often work with abstract mathematical representations of a problem
3. Best time complexity $\not\Rightarrow$ actually useful

Takeaways

1. These algorithms are for *massive n* (Toom-Cook crossover is at numbers with hundreds of digits)
2. Algorithms often work with abstract mathematical representations of a problem
3. Best time complexity $\not\Rightarrow$ actually useful
4. Shared techniques (e.g., divide-and-conquer)