Matrix Multiplication

Randomized Algorithms: Week 3
Summer HSSP 2023
Emily Liu

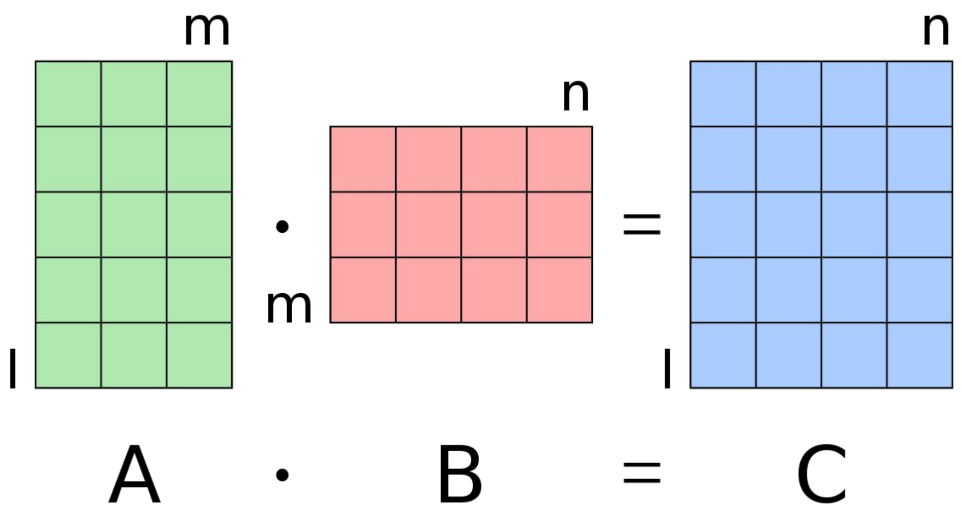
Review: Matrices

Matrix: 2D array of numbers (n rows, m columns → **n x m** matrix)

$$\begin{array}{ccc} & \text{3 columns} & \\ A = & \begin{bmatrix} -2 & 5 & 6 \\ 5 & 2 & 7 \end{bmatrix} & \text{2 rows} \end{array}$$

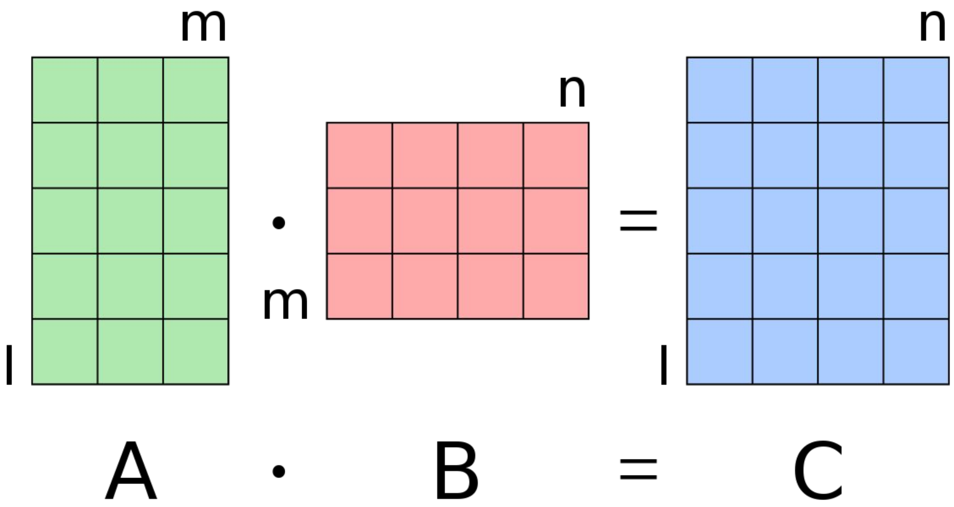
Matrix multiplication

To multiply matrices A and B, the second dimension of A must be the first dimension of B.



Matrix multiplication

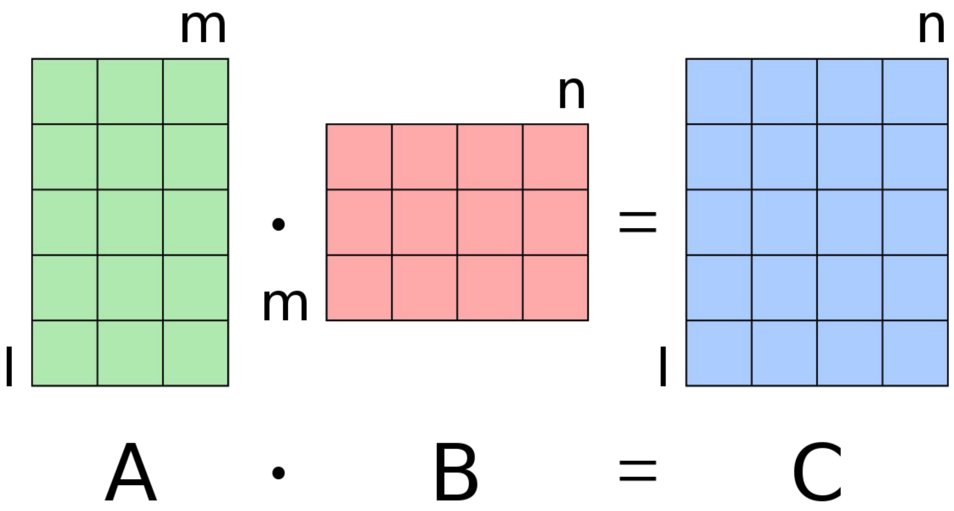
Usages: Many modern data-intensive processes, such as machine learning and dataset management, rely on matrix multiplication.



Matrix multiplication

Let C = A x B. The entry C[i][j] = sum(A[i][k] * B[k][j] for k = 0 … m-1).

- **dot product** of ith row of A, jth column of B



Matrix multiplication: Example

$$\begin{bmatrix} \mathbf{1} & \mathbf{8} \\ \mathbf{3} & \mathbf{6} \end{bmatrix} \times \begin{bmatrix} \mathbf{4} \\ \mathbf{3} \end{bmatrix}$$

Matrix multiplication: Example

$$\begin{bmatrix} 1 & 8 \\ 3 & 6 \end{bmatrix} \times \begin{bmatrix} 4 \\ 3 \end{bmatrix} = \begin{bmatrix} 1*4 + 8*3 \\ 3*4 + 6*3 \end{bmatrix}$$

$$= \begin{bmatrix} 28 \\ 30 \end{bmatrix}$$

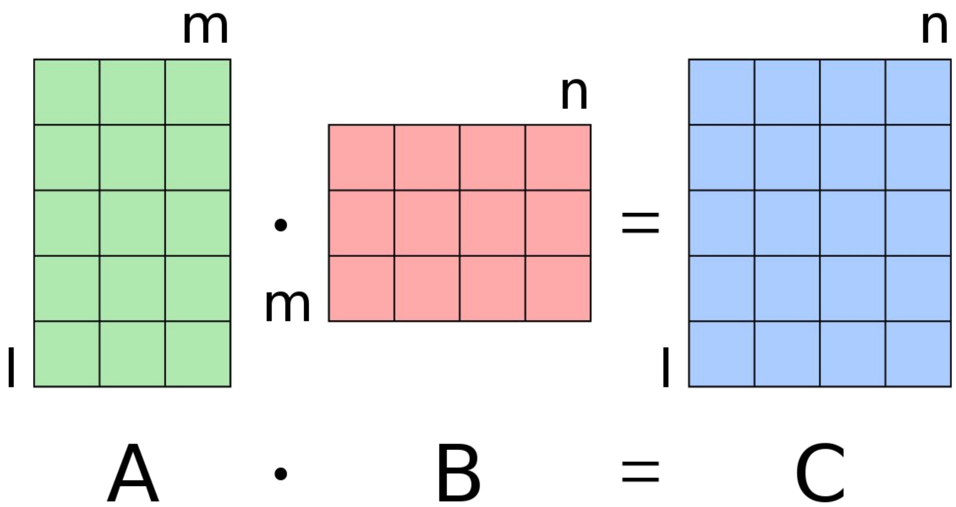
Time complexity of matrix multiplication

For each entry in C (l x n):

Need to compute the summation of m different product terms.

Therefore, overall complexity is **O(lmn)**.

For square matrices, is **O(n^3)**.



Matrix multiplication is slow…

Naive method: $\mathbf{O(n^3)}$

Strassen's method: $\mathbf{O(n^{\log 7}) \approx O(n^{2.8})}$

Williams et al (2020): $\mathbf{O(n^{2.3728596})}$

Duan et al (2022): $\mathbf{O(n^{2.371866})}$

Williams et al (2023): $\mathbf{O(n^{2.371552})}$

Changing the problem

Matrix multiplication

(given A and B, find A x B)

$\rightarrow$

Matrix multiplication **verification**

(given A, B, and C, return True if A x B = C and False otherwise.)

Applications: error-checking, ID verification

Matrix multiplication verification

Task: Given A, B, and C, return True if A x B = C and False otherwise.

Observe:

- Matrix multiplication is a **stronger** problem than matrix multiplication verification; when we solve matrix multiplication (using any of the previous methods), we can solve the verification task in $O(n^3)$ $O(n^{\log 7})$, etc…
- Randomized approach: can get to $\mathbf{O(n^2)}$.

Frievalds' Algorithm

Assume A, B, and C are all n x n matrices.

Define a length n vector **r** consisting of 0's and 1's. Each entry in **r** has a ½ probability of being 0 or 1.

Return True if the following expression is true, otherwise False.

$$A\,(B\,r) = C\,r.$$

Time complexity of Frievalds

In general: Matrix-vector multiplication is faster than matrix multiplication.

RHS: Computation of C x r: O(n * n * 1) = $O(n^2)$

LHS: Compute A x B x r

- Compute s = B x r first in $O(n^2)$
- Compute A x s in O(n^2)

Correctness of Frievalds

Case 1: A x B = C (Correct answer: True)

- Frievalds will always return True, which is correct answer!

Case 2: A x B != C (Correct answer: False)

- **Question: What is the probability Frievalds returns False?**

Correctness of Frievalds

Claim: When $AB \neq C$, $P(ABr \neq Cr) \geq ½$.

What this means:

When $AB \neq C$, Frievalds has a $\leq ½$ probability of being wrong.

Proof:

Let $D = AB - C$.

Now, we find $P(D\ r \neq 0)$.

Correctness of Frievalds

Claim: Sine $D \neq 0$, there exist indices i, j such that $D[i][j] \neq 0$.

We can select a length n vector **v** such that $v[j] = 1$ and all other entries are 0.

Observation: $D v \neq 0$.

Now consider a badly selected **r** such that $D r = 0$ (algorithm returns True).

Define: $r' = r + v$.

Observe: $D r' = D r + D v = 0 + [\text{something nonzero}] \neq 0$.

Correctness of Frievalds

Remember that $v[i] = 0$ for all $i \neq j$.

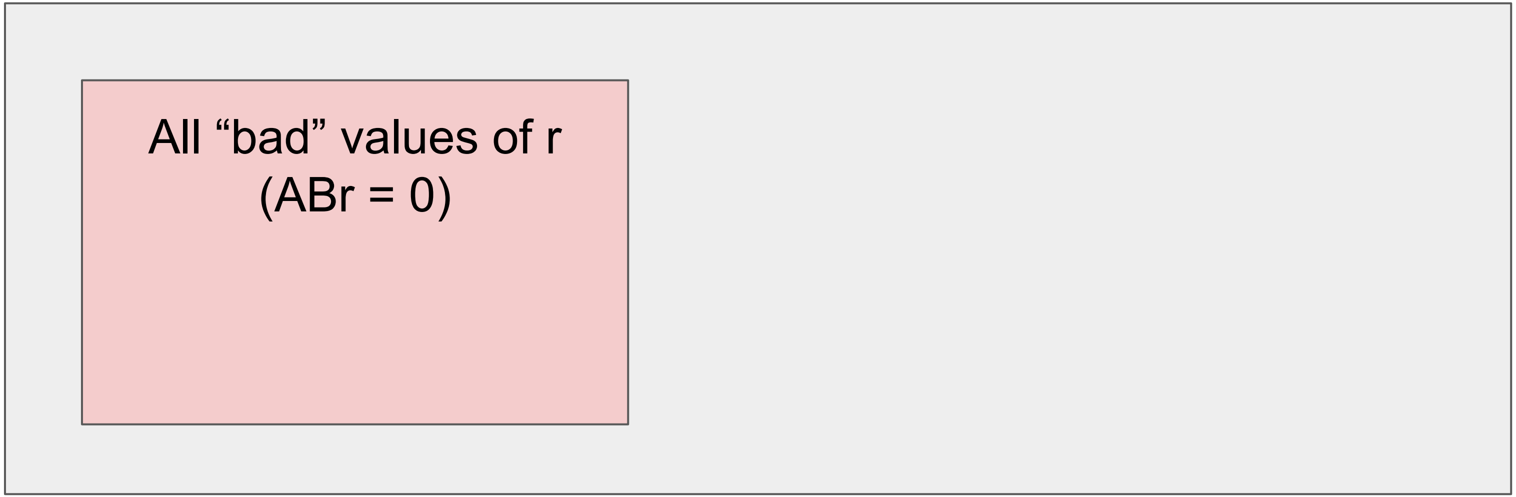
This means that $r' = r$ at all other indices that are not j.

All possible values of r

Correctness of Frievalds

Remember that v[i] = 0 for all i ≠ j.

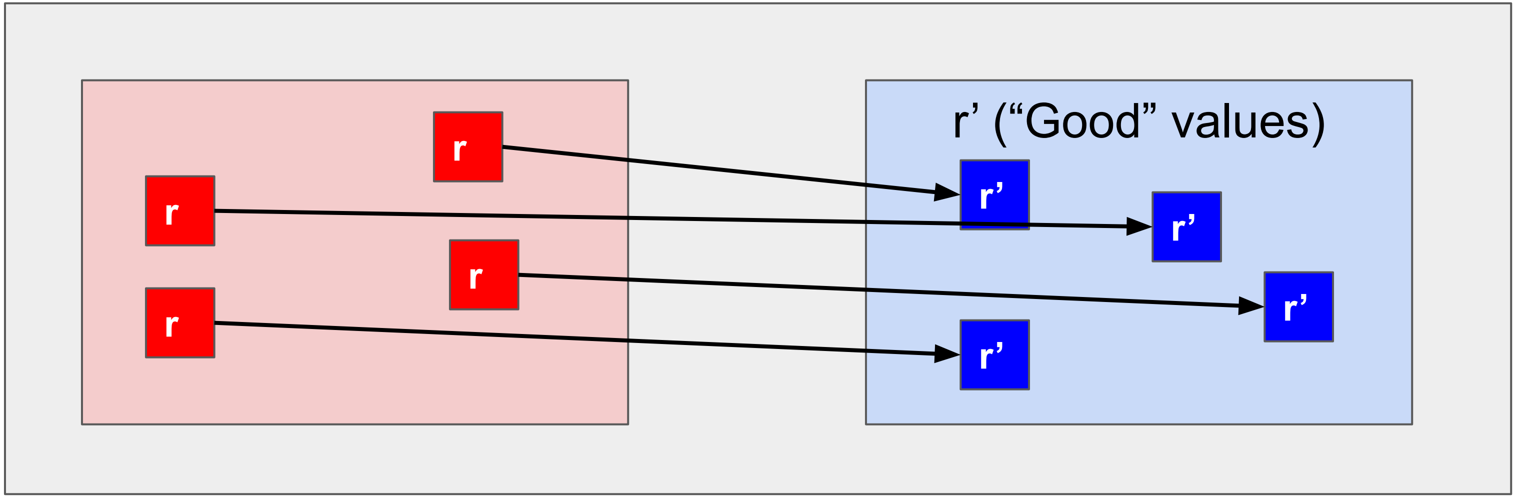
This means that r’ = r at all other indices that are not j.



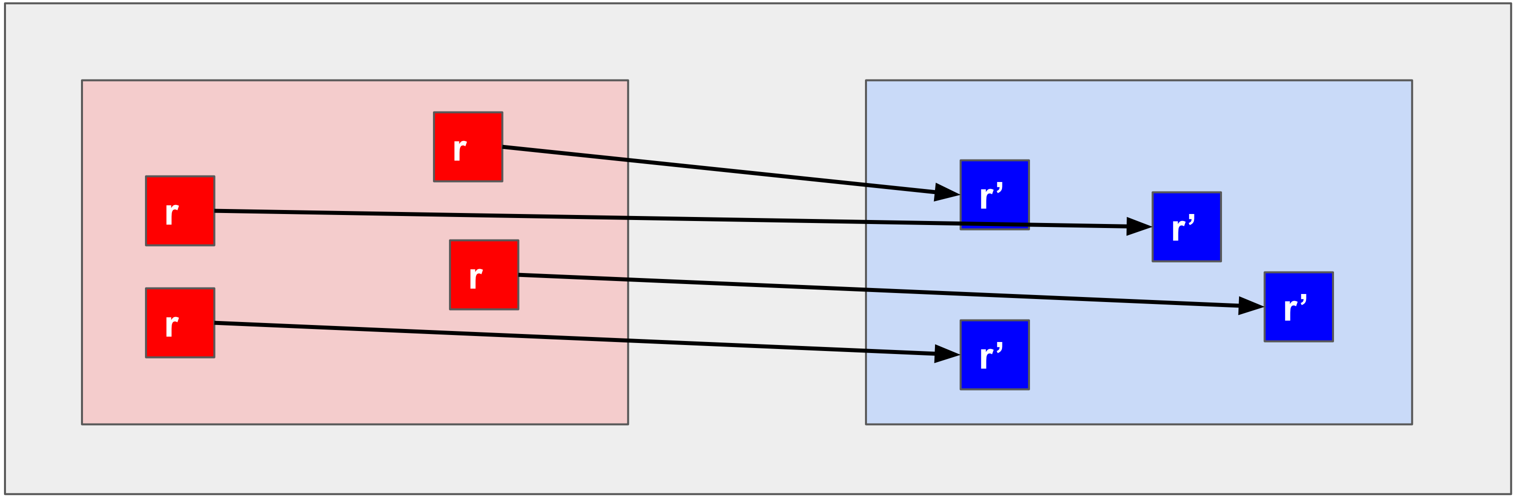
Correctness of Frievalds

Remember that v[i] = 0 for all i ≠ j.

This means that r’ = r at all other indices that are not j.



Correctness of Frievalds



For every "bad" r, such that ABr = 0, we can construct a "good" r' (that is also a valid vector we could select randomly) where ABr' ≠ 0.

Thus, P(ABr = 0) ≤ ½, and P(correct) ≥ ½.

Conclusion

Matrix multiplication (of n x n matrices) is slow; brute force is $O(n^3)$ and optimizations get to $O(n^{2.3})$ at best.

Frievalds' algorithm is a true-biased Monte Carlo algorithm that performs matrix multiplication **verification** in $O(n^2)$ time, using the fact that matrix-vector multiplication is faster than matrix-matrix multiplication.

Regardless of input, Frievalds' algorithm has at most a 50% chance of failure in the case where the algorithm should return False.

Exercises

Warmup: Matrix multiplication

1. Implement standard (brute-force) matrix multiplication in your programming language of choice.
2. (Optional) Implement Strassen’s divide-and-conquer algorithm for matrix multiplication.
    a. Resource: https://www.geeksforgeeks.org/strassens-matrix-multiplication/

Frievalds’ Algorithm

3. Implement Frievalds’ Algorithm in your programming language of choice for square matrices.
4. Try plotting the accuracy of Frievald’s vs False cases versus the number of times you run the algorithm.
5. Extend Frievalds’ Algorithm to matrix multiplication of any dimension.

Additional resources

Matrices: https://www.khanacademy.org/math/precalculus/x9e81a4f98389efdf:matrices/x9e81a4f98389efdf:mat-intro/a/intro-to-matrices

Matrix multiplication: https://www.mathsisfun.com/algebra/matrix-multiplying.html

Matmul in code: https://www.geeksforgeeks.org/c-program-multiply-two-matrices/

Frievalds implementation: https://www.geeksforgeeks.org/freivalds-algorithm/

Strassen’s method: https://www.geeksforgeeks.org/strassens-matrix-multiplication/