WIKIPEDIA

# Freivalds' algorithm

**Freivalds' algorithm** (named after Rūsiņš Mārtiņš Freivalds) is a probabilistic randomized algorithm used to verify matrix multiplication. Given three $n \times n$ matrices $A$, $B$, and $C$, a general problem is to verify whether $A \times B = C$. A naïve algorithm would compute the product $A \times B$ explicitly and compare term by term whether this product equals $C$. However, the best known matrix multiplication algorithm runs in $O(n^{2.3729})$ time.[1] Freivalds' algorithm utilizes randomization in order to reduce this time bound to $O(n^2)$ [2] with high probability. In $O(kn^2)$ time the algorithm can verify a matrix product with probability of failure less than $2^{-k}$.

## Contents



## The algorithm

### Input

Three $n \times n$ matrices $A$, $B$, and $C$.

### Output

Yes, if $A \times B = C$; No, otherwise.

### Procedure

1. Generate an $n \times 1$ random 0/1 vector $\vec{r}$.
2. Compute $\vec{P} = A \times (B\vec{r}) - C\vec{r}$.
3. Output "Yes" if $\vec{P} = (0, 0, \ldots, 0)^T$; "No," otherwise.

### Error

If $A \times B = C$, then the algorithm always returns "Yes". If $A \times B \neq C$, then the probability that the algorithm returns "Yes" is less than or equal to one half. This is called one-sided error.

By iterating the algorithm $k$ times and returning "Yes" only if all iterations yield "Yes", a runtime of $O(kn^2)$ and error probability of $\leq 1/2^k$ is achieved.

## Example

Suppose one wished to determine whether:

$$AB = \begin{bmatrix} 2 & 3 \\ 3 & 4 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 1 & 2 \end{bmatrix} \stackrel{?}{=} \begin{bmatrix} 6 & 5 \\ 8 & 7 \end{bmatrix} = C.$$

A random two-element vector with entries equal to 0 or 1 is selected — say $\vec{r} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$ — and used to compute:

$$\begin{aligned} A \times (B\vec{r}) - C\vec{r} &= \begin{bmatrix} 2 & 3 \\ 3 & 4 \end{bmatrix} \left( \begin{bmatrix} 1 & 0 \\ 1 & 2 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} \right) - \begin{bmatrix} 6 & 5 \\ 8 & 7 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} \\ &= \begin{bmatrix} 2 & 3 \\ 3 & 4 \end{bmatrix} \begin{bmatrix} 1 \\ 3 \end{bmatrix} - \begin{bmatrix} 11 \\ 15 \end{bmatrix} \\ &= \begin{bmatrix} 11 \\ 15 \end{bmatrix} - \begin{bmatrix} 11 \\ 15 \end{bmatrix} \\ &= \begin{bmatrix} 0 \\ 0 \end{bmatrix}. \end{aligned}$$

This yields the zero vector, suggesting the possibility that AB = C. However, if in a second trial the vector $\vec{r} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$ is selected, the result becomes:

$$A \times (B\vec{r}) - C\vec{r} = \begin{bmatrix} 2 & 3 \\ 3 & 4 \end{bmatrix} \left( \begin{bmatrix} 1 & 0 \\ 1 & 2 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} \right) - \begin{bmatrix} 6 & 5 \\ 8 & 7 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} -1 \\ -1 \end{bmatrix}.$$

The result is nonzero, proving that in fact AB ≠ C.

There are four two-element 0/1 vectors, and half of them give the zero vector in this case ($\vec{r} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$ and $\vec{r} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$), so the chance of randomly selecting these in two trials (and falsely concluding that AB=C) is $1/2^2$ or 1/4. In the general case, the proportion of $r$ yielding the zero vector may be less than 1/2, and a larger number of trials (such as 20) would be used, rendering the probability of error very small.

## Error analysis

Let $p$ equal the probability of error. We claim that if $A \times B = C$, then $p = 0$, and if $A \times B \neq C$, then $p \leq 1/2$.

### Case $A \times B = C$

$$\begin{aligned}\vec{P} &= A \times (B\vec{r}) - C\vec{r} \\ &= (A \times B)\vec{r} - C\vec{r} \\ &= (A \times B - C)\vec{r} \\ &= \vec{0}\end{aligned}$$

This is regardless of the value of $\vec{r}$, since it uses only that $A \times B - C = 0$. Hence the probability for error in this case is:

$$\Pr[\vec{P} \neq 0] = 0$$

### Case $A \times B \neq C$

Let

$$\vec{P} = D \times \vec{r} = (p_1, p_2, \ldots, p_n)^T$$

Where

$$D = A \times B - C = (d_{ij}).$$

Since $A \times B \neq C$, we have that some element of $D$ is nonzero. Suppose that the element $d_{ij} \neq 0$. By the definition of matrix multiplication, we have:

$$p_i = \sum_{k=1}^{n} d_{ik} r_k = d_{i1} r_1 + \cdots + d_{ij} r_j + \cdots + d_{in} r_n = d_{ij} r_j + y.$$

For some constant $y$. Using Bayes' Theorem, we can partition over $y$:

$$\Pr[p_i = 0] = \Pr[p_i = 0 | y = 0] \cdot \Pr[y = 0] + \Pr[p_i = 0 | y \neq 0] \cdot \Pr[y \neq 0] \quad (1)$$

We use that:

$$\begin{aligned}\Pr[p_i = 0 | y = 0] &= \Pr[r_j = 0] = \frac{1}{2} \\ \Pr[p_i = 0 | y \neq 0] &= \Pr[r_j = 1 \wedge d_{ij} = -y] \leq \Pr[r_j = 1] = \frac{1}{2}\end{aligned}$$

Plugging these in the equation (1), we get:

$$\begin{aligned}\Pr[p_i = 0] &\leq \frac{1}{2} \cdot \Pr[y = 0] + \frac{1}{2} \cdot \Pr[y \neq 0] \\ &= \frac{1}{2} \cdot \Pr[y = 0] + \frac{1}{2} \cdot (1 - \Pr[y = 0]) \\ &= \frac{1}{2}\end{aligned}$$

Therefore,

$$\Pr[\vec{P} = 0] = \Pr[p_1 = 0 \wedge \cdots \wedge p_i = 0 \wedge \cdots \wedge p_n = 0] \leq \Pr[p_i = 0] \leq \frac{1}{2}.$$

This completes the proof.

## Ramifications

Simple algorithmic analysis shows that the running time of this algorithm is $O(n^2)$, beating the classical deterministic algorithm's bound of $O(n^3)$. The error analysis also shows that if we run our algorithm $k$ times, we can achieve an error bound of less than $\frac{1}{2^k}$, an exponentially small quantity. The algorithm is also fast in practice due to wide availability of fast implementations for matrix-vector products. Therefore, utilization of randomized algorithms can speed up a very slow deterministic algorithm. In fact, the best known deterministic matrix multiplication algorithm known at the current time is a variant of the Coppersmith–Winograd algorithm with an asymptotic running time of $O(n^{2.3729})$.[1]

Freivalds' algorithm frequently arises in introductions to probabilistic algorithms due to its simplicity and how it illustrates the superiority of probabilistic algorithms in practice for some problems.

## See also

- Schwartz–Zippel lemma

## References


1. Virginia Vassilevska Williams. "Breaking the Coppersmith-Winograd barrier" (http://www.cs.berkeley.edu/~virgi/matrixmult-f.pdf) (PDF).
2. Raghavan, Prabhakar (1997). "Randomized algorithms" (http://portal.acm.org/citation.cfm?id=234327). *ACM Computing Surveys*. **28**. doi:10.1145/234313.234327 (https://doi.org/10.1145%2F234313.234327). Retrieved 2008-12-16.

- Freivalds, R. (1977), "Probabilistic Machines Can Use Less Running Time", IFIP Congress 1977, pp. 839–842.