The rank of 4×4 matrix multiplication over $\mathbb{F}_2$ is at least 35

Tejasvi Singh Tomar*

hexagon:2611.00372v1

Abstract

We prove that the rank of the 4×4 matrix multiplication tensor over $\mathbb{F}_2$ is at least 35; the previous lower bound was 34. The bound follows by substitution from $\mathrm{R}_{\mathbb{F}_2}(\langle 3,3,4\rangle) \geq 27$, which also gives $\mathrm{R}_{\mathbb{F}_2}(\langle 3,4,4\rangle) \geq 31$. Each of the three bounds is one more than what follows from $\mathrm{R}_{\mathbb{F}_2}(\langle 3,3,3\rangle) = 23$. A flattening argument reduces the bound for $\langle 3,3,4\rangle$ to lower bounds for two quotients of the tensor, by the span of a matrix of rank two and by the span of a matrix of rank three. These come from exhaustive searches over the possible patterns of a short decomposition, pruned by lower bounds for quotients by larger subspaces. The whole argument, including every step of the searches, is checked in the proof assistant Lean 4, with one compiled evaluation of a verified certificate checker.

Keywords: matrix multiplication, tensor rank, bilinear complexity, lower bounds, finite fields, computer-assisted proof, Lean.
MSC 2020: 68Q17, 15A69, 68V20.

1 Introduction

The rank $\mathrm{R}(\langle n,m,p\rangle)$ of the matrix multiplication tensor $\langle n,m,p\rangle$ is the least number of multiplications in a bilinear algorithm for the product of an $n \times m$ and an $m \times p$ matrix. An algorithm for a small format can be applied recursively to the blocks of large matrices, as Strassen's algorithm for 2×2 matrices is [Str69], so the ranks of small formats also bear on the multiplication of large matrices. Even for 4×4 matrices the gap between the known bounds is wide. Over $\mathbb{F}_2$ computation enters on both sides of the gap: the best algorithm, with 47 multiplications, was found by a search over $\mathbb{F}_2$ [FBH+22] (outside characteristic two the best known algorithm uses 48 [N+25, DPS25]), and since a decomposition over $\mathbb{F}_2$ is a finite object, lower bounds can be attacked by exhaustive computation. The best lower bound has been 34. This value is the case $m = n = 4$ of Bläser's inequality $\mathrm{R}(\langle n,m,n\rangle) \geq 2mn + 2n - m - 2$ for $m \geq n \geq 3$ [Blä03], and over $\mathbb{F}_2$ it also follows from the recent determination of $\mathrm{R}_{\mathbb{F}_2}(\langle 3,3,3\rangle)$, as we explain next.

Computer-assisted proofs have recently raised the lower bounds over $\mathbb{F}_2$ for several small formats. Wang [Wan26a, Theorem 1] proved $\mathrm{R}_{\mathbb{F}_2}(\langle 3,3,4\rangle) \geq 25$ and $\mathrm{R}_{\mathbb{F}_2}(\langle 3,4,4\rangle) \geq 29$, above the values 24 and 28 of Bläser's inequalities [Blä03, Blä99]. For $\langle 3,3,3\rangle$ the lower bound over $\mathbb{F}_2$ rose from 19 [Blä03] to 20 [Wan26a], 21 [Wan26b, Yan26], 22 [RR26] (proved there over $\mathbb{F}_2$ and then for integer coefficients) and 23 [Tom26], which is the rank by Laderman's algorithm [Lad76]. The substitution step of Lemma 3.3 turns $\mathrm{R}_{\mathbb{F}_2}(\langle 3,3,3\rangle) = 23$ into $\mathrm{R}_{\mathbb{F}_2}(\langle 3,3,4\rangle) \geq 26$, $\mathrm{R}_{\mathbb{F}_2}(\langle 3,4,4\rangle) \geq 30$ and $\mathrm{R}_{\mathbb{F}_2}(\langle 4,4,4\rangle) \geq 34$. We improve each of these bounds by one.

Theorem 1.1. *Over* $\mathbb{F}_2$, $\mathrm{R}(\langle 3,3,4\rangle) \geq 27$, $\mathrm{R}(\langle 3,4,4\rangle) \geq 31$ *and* $\mathrm{R}(\langle 4,4,4\rangle) \geq 35$.

*ORCID: 0000-0003-4668-3639.

In particular, 35 is the first lower bound above 34 for the rank of 4×4 matrix multiplication over $\mathbb{F}_2$. A decomposition with integer coefficients reduces modulo 2 to a decomposition over $\mathbb{F}_2$ of the same length, so the three bounds also hold for bilinear algorithms with integer coefficients. Over $\mathbb{F}_2$ the three ranks now lie between 27 and 29, between 31 and 38, and between 35 and 47; the upper bounds 29 and 38 are due to Smirnov [Smi13].

The proof starts from a flattening. Write a decomposition as $\langle 3,3,4\rangle = \sum_t a_t \otimes b_t \otimes c_t$ with a_t a 3×3 matrix. Grouping the row index of a_t with the second factor and its column index with the third turns $\langle 3,3,4\rangle$ into a 36×36 matrix of full rank, to which each term contributes at most $\mathrm{rk}(a_t)$ (Lemma 3.1). So a decomposition with at most 26 terms has a term whose first factor has rank two or three. Up to the symmetries $a \mapsto PaQ$ with P, Q invertible there are only two such matrices, v_2 and v_3, and a term with first factor v vanishes in the quotient of $\langle 3,3,4\rangle$ by $\mathrm{span}(v)$. The first bound therefore follows once each of the two quotients is shown to have rank at least 26.

These two quotient bounds are the computational core. A short decomposition of the quotient by a subspace E determines a multiset of cosets of E, its *profile*, and the profile must respect a *cap* for every larger subspace S: the number of its elements in S is at most the length minus a lower bound for the quotient by S. We enumerate every profile that meets all caps, up to the symmetries that fix E, and refute the survivors: for v_3 there are none, and for v_2 there is one. The caps need lower bounds for the quotients by larger subspaces, which we prove by induction on the dimension, about half of them by searches of the same kind.

We assume no earlier lower bound; the bounds whose values come from earlier work are proved again here. The approach builds on the recent proofs for $\langle 3,3,3\rangle$. Caps, coset profiles and the symmetry-reduced enumeration of profiles come from D'Ambrosio [D'Am26], Wang [Wan26a, Wan26b] and Yang et al. [Yan26]. The tables of subspace bounds and the forced-product argument come from Wang's framework [Wan26a], and most of the bounds for larger subspaces have the values his program computes. Yang et al. [Yan26] introduced the split flattening, of which Lemma 3.1 is the analogue for $\langle 3,3,4\rangle$; they also exclude profiles by weighted counting over covers of a quotient, with certificates made of nonnegative multipliers, and our pruning step (Section 4) and the coset counting of Section 5 are of this kind. The refutation of the surviving profiles by an exhaustive search with branching certificates follows [Tom26]; case analyses that branch on the second and third factors also occur in the Lean proof of [RR26].

The proof, with its searches and certificates, is checked in Lean 4 with Mathlib [dMU21, mC20]. Besides the Lean kernel, its trust base contains one evaluation by Lean's compiler of a certificate checker that is itself proved sound, and that evaluation must be trusted to agree with the kernel (Section 7). Section 2 fixes the notation, Section 3 reduces the theorem to the two quotient bounds, Section 4 describes the searches, Section 5 the bounds for larger subspaces, Section 6 the certificates for the leaves of the searches, Section 7 the formal proof, and Section 8 some open questions.

2 Setting

All vector spaces are over $\mathbb{F}_2$. Let $A = \mathbb{F}_2^{3\times 3}$, $B = \mathbb{F}_2^{3\times 4}$ and $C = \mathbb{F}_2^{4\times 3}$, with standard bases E_{ij}, F_{jk} and G_{ki}. The tensor is

$$T = \langle 3,3,4\rangle = \sum_{i,j<3,\ k<4} E_{ij} \otimes F_{jk} \otimes G_{ki} \in A \otimes B \otimes C,$$

and in general $\langle n,m,p\rangle = \sum_{i<n,\, j<m,\, k<p} E_{ij} \otimes F_{jk} \otimes G_{ki}$, with the matrix units of the spaces of $n \times m$, $m \times p$ and $p \times n$ matrices. A *decomposition* of length r is an equation $T = \sum_{t=1}^{r} a_t \otimes b_t \otimes c_t$, and $\mathrm{R}(T)$ is the least length. For a subspace $S \subseteq A$ let $T \bmod S = (\pi_S \otimes 1 \otimes 1)(T)$, where $\pi_S : A \to A/S$ is the quotient map. A subspace of dimension one is a *point*.

Lemma 2.1. $\mathrm{R}(T \bmod S)$ *is the least number of terms with* $a_t \notin S$*, over all decompositions of* T*.*

Proof. Applying $\pi_S \otimes 1 \otimes 1$ to a decomposition kills the terms with $a_t \in S$. Conversely, lift a shortest decomposition of $T \bmod S$ term by term; the difference from T lies in $S \otimes B \otimes C$ and is a sum of terms whose first factors lie in S. □

We write $\mathrm{QuotLB}(S, k)$ for the statement that every decomposition of T has at least k terms with first factor outside S; by Lemma 2.1 it says $\mathrm{R}(T \bmod S) \geq k$. This form needs no quotient spaces, and it is the one we formalize.

Symmetry. For $P, Q \in \mathrm{GL}_3(\mathbb{F}_2)$ the map $(a, b, c) \mapsto (PaQ,\, Q^{-1}b,\, cP^{-1})$ fixes T, so it maps decompositions to decompositions of the same length, and it maps the terms with $a_t \in S$ to the terms with $a_t \in PSQ$. Hence $\mathrm{QuotLB}(S, k)$ and $\mathrm{QuotLB}(PSQ, k)$ are equivalent. We write G_0 for this group acting on A; it has order 28,224. The map $(a, b, c) \mapsto (a^{\mathsf{T}}, c^{\mathsf{T}}, b^{\mathsf{T}})$ is also a symmetry of T; we do not use it, since the arguments below work with any group of symmetries. The nonzero matrices fall into three G_0-orbits, by rank, with 49, 294 and 168 elements; we fix the representatives

$$v_2 = E_{01} + E_{10} \quad (\text{rank } 2), \qquad v_3 = E_{02} + E_{11} + E_{20} \quad (\text{rank } 3).$$

3 Reduction to two quotients

Lemma 3.1 (split flattening)**.** *Every decomposition* $T = \sum_t a_t \otimes b_t \otimes c_t$ *satisfies* $\sum_t \mathrm{rk}(a_t) \geq 36$*.*

Proof. Identify A with $U \otimes V$, where $U = V = \mathbb{F}_2^3$ carry the row and column index, and regroup $A \otimes B \otimes C$ as $(U \otimes B) \otimes (V \otimes C)$. This regrouping is linear and turns each tensor into a 36×36 matrix with rows indexed by pairs (i, y) and columns by pairs (j, z), where y and z run over the standard bases of B and C. The image of T is $\sum_{i,j,k} (e_i \otimes F_{jk}) \otimes (e_j \otimes G_{ki})$; since $(i, j, k) \mapsto (i, (j, k))$ and $(i, j, k) \mapsto (j, (k, i))$ are both bijections onto $[3] \times [12]$, it is a permutation matrix, of rank 36. The image of $a \otimes b \otimes c$ is the Kronecker product of a with the rank-one matrix bc^{T}, of rank at most $\mathrm{rk}(a)$. Subadditivity of rank gives the claim. □

The choice of pairing matters, since grouping the row index of a with C instead gives a matrix of rank only 4.

Proposition 3.2. *If* $\mathrm{QuotLB}(\mathrm{span}(v_2), 26)$ *and* $\mathrm{QuotLB}(\mathrm{span}(v_3), 26)$ *hold, then* $\mathrm{R}(T) \geq 27$*.*

Proof. Take a decomposition with at most 26 terms. By Lemma 3.1 the ranks of the first factors sum to at least 36, so some a_{t_0} has rank two or three. Apply the symmetry (P, Q) that sends a_{t_0} to v_2 or v_3. In the new decomposition the first factor of term t_0 lies in $\mathrm{span}(v)$, so at most 25 first factors lie outside it, which contradicts the hypothesis for v. □

The two hypotheses are proved in Sections 4–6. The other two bounds follow by the substitution method [Pan66]; we recall the step in the form we use.

Lemma 3.3. *If* $\langle n, m, p+1 \rangle$ *has a decomposition of length* $r + m$*, then* $\langle n, m, p \rangle$ *has one of length at most* r*. Moreover* $\mathrm{R}(\langle n, m, p \rangle) = \mathrm{R}(\langle m, p, n \rangle)$*.*

Proof. Write $\langle n, m, p+1 \rangle = \sum_{j,k} F_{jk} \otimes M_{jk}$ with $M_{jk} = \sum_i E_{ij} \otimes G_{ki}$. The m slices $M_{j,p}$ are linearly independent, so in any decomposition there are m terms whose second factors restrict to a basis of the coordinates $F_{\cdot,p}$; together with the F_{jk}, $k < p$, these m second factors form a basis of the second factor space, so the following linear map exists. The linear map of the second factor

space that fixes every F_{jk} with $k < p$ and sends these m second factors to zero removes those m terms and adds multiples of the $M_{j',p}$ to the other slices. Those multiples involve only the last row of the third matrix, and setting that row to zero is a linear map that turns the result into a decomposition of $\langle n, m, p\rangle$ of length at most r. The cyclic statement holds because $(a, b, c) \mapsto (b, c, a)$ maps decompositions of $\langle n, m, p\rangle$ to decompositions of $\langle m, p, n\rangle$. □

Proof of Theorem 1.1, given the two quotient bounds. Proposition 3.2 gives $\mathrm{R}(\langle 3, 3, 4\rangle) \geq 27$. If $\langle 3, 4, 4\rangle$ had a decomposition of length $30 = 26 + 4$, then Lemma 3.3 would give one of $\langle 3, 4, 3\rangle$ of length at most 26, and by cyclic invariance one of $\langle 3, 3, 4\rangle$; so $\mathrm{R}(\langle 3, 4, 4\rangle) \geq 31$. In the same way a decomposition of $\langle 4, 4, 4\rangle$ of length 34 would give one of $\langle 3, 4, 4\rangle$ of length at most 30, so $\mathrm{R}(\langle 4, 4, 4\rangle) \geq 35$. □

4 Lower bounds for quotients by a search

Fix a subspace $E \neq A$ and a length L. To prove $\mathrm{QuotLB}(E, L + 1)$ we suppose that some decomposition of T has at most L terms with first factor outside E. Two simple observations restrict what such a decomposition can look like.

Lemma 4.1 (padding)**.** *If some decomposition of T has at most L terms outside E, then some decomposition has exactly L terms outside E, and in each of them a_t, b_t, c_t are nonzero.*

Proof. Drop zero terms. Some term lies outside E, because a functional on A that vanishes on E but not on all of A sees a nonzero slice of T. While fewer than L terms are outside E, split one of them as $a \otimes b \otimes c = a \otimes b \otimes c' + a \otimes b \otimes (c + c')$ with $c' \notin \{0, c\}$. □

Lemma 4.2 (caps)**.** *Let $E \subsetneq S \subseteq A$, and let $\mathrm{lb}(S)$ be a number for which $\mathrm{QuotLB}(S, \mathrm{lb}(S))$ holds. In a decomposition with exactly L terms outside E, at most $L - \mathrm{lb}(S)$ of those terms have $a_t \in S$.*

Proof. Every term outside S is outside E, so the number of terms outside S is L minus the number of outside terms in S, and it is at least $\mathrm{lb}(S)$. □

The *profile* of such a decomposition is the multiset of the cosets $a_t + E$ of its L outside terms. By Lemma 4.2, for every strict superspace S of E at most $\mathrm{cap}(S) = L - \mathrm{lb}(S)$ elements of the profile lie in S/E. For a point E there are 417,198 strict superspaces. Section 5 explains where their bounds come from.

Enumeration. Number the nonzero cosets of E (255 of them when E is a point), and write a profile as the nondecreasing sequence of the numbers of its elements. Let H be a subgroup of the stabilizer of E in G_0; for the two points we take the whole stabilizer, of order 96 for v_2 and 168 for v_3, and for the searches of Section 5 the stabilizer or, when that is large, its subgroup that also fixes a few cosets. The group H permutes the cosets, and we call a sorted sequence *canonical* if no element of H maps it to a lexicographically smaller sorted sequence.

Lemma 4.3. *Every prefix of a canonical sequence is canonical.*

Proof. Let $s = (s_1 \leq \cdots \leq s_L)$ be canonical, and suppose that some $h \in H$ maps a prefix $(s_1, \ldots, s_k)$ to a sorted sequence $(u_1 \leq \cdots \leq u_k)$ that is lexicographically smaller. Let i be the first index with $u_i \neq s_i$, so that $u_i < s_i$. For $j \leq k$ the multiset $h(s)$ contains $u_1, \ldots, u_j$, so its j-th smallest element is at most u_j. Hence the sorted form of $h(s)$ is at most s_j at every position $j < i$ and smaller than s_i at position i, so it is lexicographically smaller than s, which contradicts the canonicity of s. □

The search builds sorted sequences one element at a time. It discards a sequence if some element of H maps it to a lexicographically smaller sorted sequence. It also discards a sequence when the caps leave no room to complete it. Suppose that ρ elements remain to be chosen, and give each coset γ that may still be chosen (no coset before the last chosen one may) a variable x_γ, the number of its further copies, with $0 \le x_\gamma \le u_\gamma$. The bound u_γ starts as the cap of $E + \langle\gamma\rangle$, decreases as copies of γ are chosen, drops to 0 when some cap containing γ is full, and can be lowered further by earlier pruning steps; in every case no completion uses more than u_γ further copies of γ. Let b_S be the room that the cap of S leaves. A completion gives a solution of $\sum_\gamma x_\gamma = \rho$ and $\sum_{\gamma \subseteq S} x_\gamma \le b_S$ for all S. For every vector $y \ge 0$ indexed by the superspaces S, every such solution satisfies

$$\rho = \sum_\gamma x_\gamma \le \sum_S y_S b_S + \sum_\gamma u_\gamma \max\Big(0,\, 1 - \sum_{S \supseteq \gamma} y_S\Big),$$

since $\sum_\gamma x_\gamma \sum_{S \supseteq \gamma} y_S = \sum_S y_S \sum_{\gamma \subseteq S} x_\gamma \le \sum_S y_S b_S$ and $x_\gamma(1 - \sum_{S \supseteq \gamma} y_S) \le u_\gamma \max(0, 1 - \sum_{S \supseteq \gamma} y_S)$. So a vector y for which the right side is smaller than ρ proves that no completion exists. The search obtains y from the dual of the linear relaxation, solved by a dense simplex method with HiGHS [HH18] as a fallback, scales it to integers and records it, and the inequality is checked in exact integer arithmetic, so a pruning step can be rechecked without solving anything. Inequalities of this kind, with recorded nonnegative multipliers, are used by Yang et al. [Yan26, Sections 3.2–3.3]. Here the multipliers come from a linear-programming solver.

Coverage and leaves. The enumeration is cut into jobs, and each job records its part of the search tree, with every branching decision and every dual vector, as a block of a proof trace (the Lean copy of each block adds the group element behind each discard). A trace proves that no further profile exists only if its blocks together cover the whole search tree, that is, every subtree that one block delegates is the root of another block. A canonical sequence of length L that meets all caps is a *leaf*. For each leaf one must decide whether T mod E has a decomposition whose first factors are the cosets of the profile, one term for each element of the profile. Restricting T to the functionals that vanish on E turns this *leaf question* into a question about L pairs of vectors (b_t, c_t) in $\mathbb{F}_2^{12} \times \mathbb{F}_2^{12}$, whose sums of outer products must match the slices of T. Section 6 refutes the leaf questions with certificates, and Table 1 gives the size of the searches.

Proposition 4.4. *Let $E \ne A$, let H be a subgroup of the stabilizer of E in G_0, and suppose that every cap comes from a valid bound* $\mathrm{lb}(S)$*. If the trace of the search covers its search tree and no leaf question has a solution, then* $\mathrm{QuotLB}(E, L+1)$ *holds.*

Proof. Otherwise Lemma 4.1 gives a decomposition with exactly L terms outside E, all with nonzero factors. Its profile respects every cap by Lemma 4.2. Some $h \in H$ maps the profile to a canonical one. Applying h to the decomposition gives a decomposition with exactly L terms outside $h(E) = E$ whose profile is the canonical one, so by Lemma 4.2 that profile also respects every cap. By Lemma 4.3 no prefix of the canonical profile is discarded for symmetry, and no prefix is discarded by the caps, so by coverage the search reaches it as a leaf. The transformed decomposition solves that leaf question, a contradiction. □

The two point searches take $L = 25$. The search for a larger subspace E in Section 5 takes $L = \mathrm{lb}(E) - 1$. On a machine with 180 virtual cores, each point search finished in about half an hour.

search	searches	trace blocks	nodes	leaves
$E = \mathrm{span}(v_3)$, $L = 25$	1	137,442	12,833,599	0
$E = \mathrm{span}(v_2)$, $L = 25$	1	7,296	1,529,941	1
larger subspaces	367	55,615	15,162,772	14,121

Table 1: The searches. Nodes are counted as recorded in the proof traces, where the root of a delegated subtree appears in two blocks; leaves are counted once each. Of the 367 searches for larger subspaces, 48 have leaves; the largest, for a subspace of dimension 3, has 2,804,206 nodes and 5,440 leaves.

5 Bounds for larger subspaces

The caps need lower bounds $\mathrm{lb}(S)$ for the quotients by subspaces of dimension 2 to 9. Up to G_0 there are 706 classes of such subspaces. Four of them consist of subspaces of matrices of rank at most one, so none of their members contains v_2 or v_3; the two point searches need the other 702. For 628 of these the value of $\mathrm{lb}(S)$ is the bound that Wang's program computes [Wan26a]. (Wang's entries impose d linear constraints on the first input; substituting them into the tensor is the reduction modulo the span S of the constraint vectors, so an entry with d constraints corresponds to a class of subspaces of dimension d here.) For the other 74 we prove higher values, first suggested by bounds for quotients of $\langle 3,3,3\rangle$ [RR26, Tom26] or found by our own searches. Each value is proved by one of five kinds of argument. Table 2 shows how they are distributed over the dimensions.

1. *Flattening.* Each term contributes rank at most one to each of the three flattenings of T mod S; for 18 classes the largest flattening rank is the bound.

2. *Forced products.* For 21 classes we use Wang's forced-product argument [Wan26a]. Let $T' = T \bmod S$, and let $\varphi_1, \dots, \varphi_f$ be functionals on one of the factors, say B, whose slices $T'(\varphi_i)$ are linearly independent. Extend them to a basis $\varphi_1, \dots, \varphi_{12}$ of B^* with dual basis $\beta_1, \dots, \beta_{12}$. In a decomposition of T' of length r, some f terms $t_1, \dots, t_f$ satisfy $\det(\varphi_i(b_{t_j})) \neq 0$. The linear map of B that kills $b_{t_1}, \dots, b_{t_f}$ and fixes $\beta_{f+1}, \dots, \beta_{12}$ sends each β_i with $i \leq f$ to $\sum_{l>f} \alpha_{il}\beta_l$ for some matrix α, and it turns the decomposition into one of length $r - f$ of $T'_\alpha = \sum_{l>f} \beta_l \otimes (T'(\varphi_l) + \sum_{i\leq f} \alpha_{il} T'(\varphi_i))$; this is a substitution step in the style of Hopcroft and Kerr [HK71]. A flattening of T'_α, or of the tensor formed by some of its slices, has the form $M(\alpha) = M_0 + \sum_m \alpha_m N_m$, so $r \geq f + \min_\alpha \mathrm{rk}\, M(\alpha)$, and it suffices that $M(\alpha)$ has rank at least $\mathrm{lb}(S) - f$ for every α. Wang checks this by enumerating the values of α. We certify it with witnesses instead: a set J of $\mathrm{lb}(S) - f$ columns and, for every nonzero vector z supported on J, a vector y_z with $y_z^{\mathsf{T}} M_0 z = 1$ and $y_z^{\mathsf{T}} N_m z = 0$ for all m. Then $y_z^{\mathsf{T}} M(\alpha) z = 1$, so $M(\alpha) z \neq 0$ for every α and every such z, and the columns in J stay linearly independent (some classes use the same condition with rows and columns exchanged). The factor, the functionals, the flattening and the witnesses for each class are part of the formal proof.

3. *Coset counting.* One class of subspaces S of dimension 6 has its bound from the seven subspaces $S + \langle c\rangle$, one for each nonzero coset c of S, which lie in three classes of dimension 7. Every matrix outside S lies in exactly one of them, so in a decomposition with r terms outside S each of these terms lies outside six of the seven subspaces, and $6r \geq \sum_c \mathrm{lb}(S + \langle c\rangle)$.

4. *Transport.* For 295 classes a symmetry in G_0 maps the subspace into a subspace of larger dimension whose bound is known. A first factor outside the larger subspace is outside the smaller one, so the bound transfers.

5. *Search.* The remaining 367 classes get their bounds from searches as in Section 4, with E the subspace itself.

A search for a subspace of dimension d uses only bounds for subspaces of dimension greater than d, and transport and coset counting also go to larger dimensions. The bounds are therefore proved by induction on the dimension, downward from the full space.

dimension	flattening	forced products	coset counting	transport	search	total
2					15	15
3				48	45	93
4				88	151	239
5		2		128	109	239
6	5	14	1	31	44	95
7	9	5			3	17
8	3					3
9	1					1
total	18	21	1	295	367	702

Table 2: The 702 classes of subspaces whose bounds the two point searches use, by dimension and by the argument that proves the bound.

6 Certificates for the leaves

A leaf question asks for nonzero $b_t, c_t \in \mathbb{F}_2^{12}$, one pair for each element $a_t + E$ of the profile, with $T(X) = \sum_{t:\, X(a_t)=1} b_t c_t^{\mathsf{T}}$ for every functional X that vanishes on E. The certificates rest on three facts about a sum of rank-one matrices.

Lemma 6.1. *Let $R = \sum_{t\in K} b_t c_t^{\mathsf{T}}$ with $u = |K|$ and all b_t, c_t nonzero.*

(i) $\operatorname{rk} R \le u$.

(ii) If $\operatorname{rk} R = u$, then every b_t lies in the column space of R, every c_t lies in its row space, and for every x with $Rx = b_t$ one has $c_t^{\mathsf{T}} x = 1$ and $c_s^{\mathsf{T}} x = 0$ for $s \neq t$; dually, $y^{\mathsf{T}} b_s$ is 1 for $s = t$ and 0 otherwise whenever $y^{\mathsf{T}} R = c_t^{\mathsf{T}}$.

(iii) If $\operatorname{rk} R \ge u - 1$, then no $t \in K$ has both b_t outside the column space of R and c_t outside its row space.

Proof. (i) is subadditivity. For (ii), write $R = BC^{\mathsf{T}}$, where the columns of B are the b_t and those of C are the c_t. Since $u = \operatorname{rk} R \le \min(\operatorname{rk} B, \operatorname{rk} C) \le u$, both B and C have linearly independent columns, so the column space of R is that of B and the row space of R is the column space of C. If $Rx = b_t$, then $B(C^{\mathsf{T}}x) = Be_t$, so $C^{\mathsf{T}}x = e_t$, which is the claim; if $y^{\mathsf{T}}R = c_t^{\mathsf{T}}$, then $(y^{\mathsf{T}}B - e_t^{\mathsf{T}})C^{\mathsf{T}} = 0$, so $y^{\mathsf{T}}B = e_t^{\mathsf{T}}$. For (iii), if b_t and c_t were outside the column and row spaces, then $\operatorname{rk}(R + b_t c_t^{\mathsf{T}}) = \operatorname{rk} R + 1 \ge u$; but $R + b_t c_t^{\mathsf{T}} = \sum_{s\neq t} b_s c_s^{\mathsf{T}}$ has rank at most $u - 1$. □

We refute each of the 14,122 distinct leaves by a branching certificate, which fixes the pairs (b_t, c_t) one term at a time. At a node, let R_X be what remains of $T(X)$ after the fixed pairs are subtracted, and apply Lemma 6.1 to R_X and the terms not yet fixed with $X(a_t) = 1$. A node where

(i) fails for some X is closed, and (ii) and (iii) restrict the column and row spaces available to the next pair. These rules leave a short list of candidates for that pair. A node of the certificate names the term to be fixed, the functionals to which (ii) and (iii) are applied, whether the symmetry reduction is used and whether the candidates are listed by b or by c first, followed by one subtree for each candidate; the checker computes the list of candidates itself, so a certificate cannot leave a candidate out. The maps $b \mapsto bW$, $c \mapsto W^{-1}c$ with $W \in \mathrm{GL}_4(\mathbb{F}_2)$ fix every slice and reduce the first branching. A certificate is checked by a program that replays the branching and verifies each rank claim by elimination. A search records a leaf each time a job reaches it, and a job that ran past its budget and was redone in smaller pieces could record a leaf twice. So 5,440 of the 19,562 certificate records repeat a leaf that already has one, 3,696 of them in the largest search and one in the search for v_2. Every record is checked.

The single leaf of the search for v_2 is a small example of the method. Its profile consists of 25 cosets of $\mathrm{span}(v_2)$, 23 of them distinct; Table 3 lists a representative of each. Its certificate has 14 nodes. At the root it branches on the term whose first factor is E_{02}. Two functionals X are tight there, with $\mathrm{rk}\, R_X = u_X = 8$ and $\mathrm{rk}\, R_X = u_X = 12$, and Lemma 6.1(ii) together with the maps $b \mapsto bW$, $c \mapsto W^{-1}c$ leaves 13 candidates for the pair of that term. For each candidate the certificate names one more term, with first factor `101/000/101`, `111/000/111` or `010/011/111`, up to three tight functionals and, in ten of the branches, one functional X with $\mathrm{rk}\, R_X = u_X - 1$; by (ii) and (iii) they leave no candidate for the pair of that term, so every branch closes at depth one. The checker verifies 31 rank claims in all. The leaves of the larger searches need far larger certificates (Appendix A).

`010/000/000` (×2)	`110/000/000` (×2)	`001/000/000`	`011/000/000`	`010/010/000`	`001/001/000`
`011/011/000`	`000/000/100`	`110/000/100`	`000/000/010`	`010/010/010`	`001/000/001`
`000/001/001`	`000/000/101`	`101/000/101`	`010/001/101`	`111/001/101`	`011/000/011`
`000/011/011`	`000/000/111`	`111/000/111`	`010/011/111`	`101/011/111`	

Table 3: The profile of the single leaf of the search for $v_2 = E_{01} + E_{10}$: one representative of each coset, as a 3×3 matrix written row by row (`010/000/000` is E_{01}), with the multiplicity where it is two.

7 Formal verification

The definitions of tensor rank and of $\langle n, m, p \rangle$ over $\mathbb{F}_2$ (as `ZMod 2`), and the three claims, are stated in one short file, with tensors as functions on finite index sets. The claims and the definitions they depend on read

```
def outer3 (a : Fin n₀ → R) (b : Fin n₁ → R) (c : Fin n₂ → R) :
    Fin n₀ → Fin n₁ → Fin n₂ → R :=
  fun i j k => a i * b j * c k
def RankLE (T : Fin n₀ → Fin n₁ → Fin n₂ → R) (r : ℕ) : Prop :=
  ∃ (a : Fin r → Fin n₀ → R) (b : Fin r → Fin n₁ → R) (c : Fin r → Fin n₂ → R),
    T = ∑ s, outer3 (a s) (b s) (c s)
def mmT (n m p : ℕ) : Fin (n * m) → Fin (m * p) → Fin (p * n) → ZMod 2 :=
  fun x y z =>
    if x.val % m = y.val / p ∧ y.val % p = z.val / n ∧ z.val % n = x.val / m
    then 1 else 0
abbrev T334 : Fin (3 * 3) → Fin (3 * 4) → Fin (4 * 3) → ZMod 2 := mmT 3 3 4
def Rank334 : Prop := ¬ RankLE T334 26
def Rank344 : Prop := ¬ RankLE (mmT 3 4 4) 30
def Rank444 : Prop := ¬ RankLE (mmT 4 4 4) 34
```

with `R` a commutative semiring; in `mmT` the entries (i, j), (j, k) and (k, i) of the three matrices have the indices $x = mi + j$, $y = pj + k$ and $z = nk + i$. The main theorems `rank334_ge_27`, `rank344_ge_31` and `rank444_ge_35` prove `Rank334`, `Rank344` and `Rank444`. As a guard against a definitional slip, the formalization also proves that $\langle 3, 3, 4\rangle$ does have a decomposition of length 29, and that the tensor agrees with the trace form $\mathrm{tr}(XYZ)$. The development uses Lean 4.30.0 and Mathlib v4.30.0.

The lemmas and propositions of Sections 3 to 6, and the symmetry statements of Section 2 that they use, are ordinary Lean proofs: the split flattening, the two-case argument and the substitution step, padding and caps, Lemma 4.3 and Proposition 4.4 for every subspace searched, the five kinds of argument of Section 5 with the induction on the dimension, and Lemma 6.1. Lemma 2.1 is not needed, since the formal statements use QuotLB. The Lean kernel decides the following computations: the flattening and forced-product certificates, the transport witnesses, every block of every search trace with its dual vectors and group elements, the coverage of each search, the tables that encode the caps, and the matching of each leaf with its certificate record. The certificate checker of Section 6 is a Lean function with a proof that every certificate it accepts refutes its leaf, and the kernel runs it on 18,751 of the 19,562 records.

The remaining 811 records, from eight of the searches for subspaces of dimension 3, are those that try the most candidates, and in the kernel they would need far more memory than the others (Appendix A). All 811 are checked in one evaluation of the same verified checker by Lean's compiler (`native_decide`). This adds exactly one axiom, `B334.T2.nat_ok._native.native_decide.ax_1_1`, to the final theorems, which therefore depend only on `propext`, `Classical.choice`, `Quot.sound` and that axiom. The axiom asserts that the compiled evaluation of one Boolean expression returned true. Trusting it means trusting that Lean's code generator, its evaluator and the runtime's arithmetic on natural numbers compute the checker as the kernel would. None of the project's own definitions carries an `implemented_by` or `extern` attribute: a scan of all 42,559 modules finds none, and finds `native_decide` used in this one module only. Compiled by the same Lean compiler into a standalone program, the checker accepts all of these certificates, and on 2,150 mutated certificates, 53 of which are still valid, its verdicts agree with those of an independently written checker.

Large constants such as the search blocks are loaded from data files at elaboration time as ordinary definitions, so a wrong data file can stop the build but cannot produce a false theorem. The statement file and the files that state the main theorems elaborate with automatic implicit arguments disabled. Appendix A gives the size of the formalization and of its build.

Before the formalization, the traces and the certificates were checked by programs written separately from the search code: a trace checker that recomputes every pruning step and the coverage, and a certificate checker for the leaves. As a positive control for the leaf rules, we moved a known decomposition of $\langle 3, 3, 4\rangle$ of length 29 into the leaf questions of 38 subspaces of dimension 3, where it gives a solution, and confirmed that along the path to that solution every branching step keeps it among the candidates.

8 Open questions

Is $\mathrm{R}_{\mathbb{F}_2}(\langle 3, 3, 4\rangle)$ equal to 27, 28 or 29? Raising the lower bound to 28 by the same route would need both point quotients to have rank at least 27, that is, the searches of Section 4 with $L = 26$, in which every cap is one larger.

Can the method work over other fields? Our computations are specific to $\mathbb{F}_2$: over a larger finite field the searches would be larger, and over an infinite field there would be nothing to enumerate.

Can the kernel check every certificate? The 811 certificates that the compiler now evaluates would need either far more kernel memory (Appendix A) or a checker designed for the kernel.

A Resources of the formal proof

Table 4 gives the size of the formalization and of its reference build, and the measurements behind the choice of the 811 certificate records that the compiler evaluates. A cost model calibrated on 8 module builds predicts more than 7,600 MiB of kernel memory, the limit for which the kernel modules of the other certificates were sized, for 490 of these records (all of them among the 513 that try more than 1,000 candidates), and about 542 GiB for the largest. The model over-predicted every one of its calibration builds, so these figures are likely overestimates. Checking these records in the kernel is a question of memory and of checker design.

modules of the formalization	42,559
CPU time of the reference build	about 120 core-hours
largest memory of one module	about 15 GiB
candidates tried by each compiled record	more than 500
compiled records that try more than 1,000 candidates	513
candidates tried by the largest compiled record	78,379
kernel memory per candidate tried, in probes with 169 to 2,853 tries	about 4 to 8 MiB

Table 4: The size of the formal proof and of its build.

Data availability

The Lean formalization, the data files it reads and the logs of the reference build are archived at `https://doi.org/10.5281/zenodo.23749574`.

Use of AI tools

Generative AI tools were used to find and prove the results, to write the programs and the Lean formalization, and to draft the text.

References

[Blä99] M. Bläser, *Lower bounds for the multiplicative complexity of matrix multiplication*, Comput. Complexity **8** (1999), 203–226.

[Blä03] M. Bläser, *On the complexity of the multiplication of matrices of small formats*, J. Complexity **19** (2003), 43–60.

[D'Am26] I. D'Ambrosio, *The exact $\mathbb{F}_2$-rank of 2×3 by 3×4 matrix multiplication*, Zenodo, 2026, `https://doi.org/10.5281/zenodo.21895176`.

[dMU21] L. de Moura and S. Ullrich, *The Lean 4 theorem prover and programming language*, in: Automated Deduction – CADE 28, Lecture Notes in Comput. Sci. 12699, Springer, 2021, 625–635.

[FBH+22] A. Fawzi et al., *Discovering faster matrix multiplication algorithms with reinforcement learning*, Nature **610** (2022), 47–53.

[HH18] Q. Huangfu and J. A. J. Hall, *Parallelizing the dual revised simplex method*, Math. Program. Comput. **10** (2018), 119–142.

[HK71] J. E. Hopcroft and L. R. Kerr, *On minimizing the number of multiplications necessary for matrix multiplication*, SIAM J. Appl. Math. **20** (1971), 30–36.

[Lad76] J. D. Laderman, *A noncommutative algorithm for multiplying* 3×3 *matrices using* 23 *multiplications*, Bull. Amer. Math. Soc. **82** (1976), 126–128.

[DPS25] J.-G. Dumas, C. Pernet and A. Sedoglavic, *A non-commutative algorithm for multiplying* 4×4 *matrices using* 48 *non-complex multiplications*, arXiv:2506.13242, 2025.

[mC20] The mathlib Community, *The Lean mathematical library*, in: Proceedings of the 9th ACM SIGPLAN International Conference on Certified Programs and Proofs (CPP 2020), 2020, 367–381.

[N+25] A. Novikov et al., *AlphaEvolve: a coding agent for scientific and algorithmic discovery*, arXiv:2506.13131, 2025.

[Pan66] V. Y. Pan, *Methods of computing values of polynomials*, Russian Math. Surveys **21** (1966), 105–136.

[RR26] I. Rudich and L.-M. Rousseau, *Lower bound of* 22 *for* 3×3 *matrix multiplication over* $\mathbb{Z}$, arXiv:2610.01639, 2026.

[Smi13] A. V. Smirnov, *The bilinear complexity and practical algorithms for matrix multiplication*, Comput. Math. Math. Phys. **53** (2013), 1781–1795.

[Str69] V. Strassen, *Gaussian elimination is not optimal*, Numer. Math. **13** (1969), 354–356.

[Tom26] T. S. Tomar, *The rank of* 3×3 *matrix multiplication over* $\mathbb{F}_2$ *is* 23, arXiv:2610.11823, 2026.

[Wan26a] C. Wang, *Automated lower bounds for bilinear complexity over finite fields*, arXiv:2603.07280, 2026.

[Wan26b] C. Wang, *A lower bound of* 21 *for* 3×3 *matrix multiplication over* $\mathbb{F}_2$, arXiv:2609.06725, 2026.

[Yan26] S. Yang et al., *A structural proof of the lower bound* 21 *for* 3×3 *matrix multiplication over* $\mathbb{F}_2$, arXiv:2609.18722, 2026.