

Exploring the Meta Flip Graph for Matrix Multiplication¹

Manuel Kauers

Institute for Algebra · Johannes Kepler University, Linz, Austria

Isaac Wood

Institute for Algebra · Johannes Kepler University, Linz, Austria

Abstract

Continuing recent investigations of bounding the tensor rank of matrix multiplication using flip graphs, we present here improved rank bounds for 24 matrix formats.

Keywords: Matrix multiplication, Tensor rank, Flip graphs.

1. Introduction

During the past few years, we have seen renewed activity in the search for matrix multiplication schemes of low rank for small and specific matrix multiplication formats. After Strassen's discovery that two 2×2 matrices over any coefficient ring can be multiplied using only seven multiplications, one less than the standard algorithm, people have already been searching intensively for multiplication schemes that multiply an $n \times m$ matrix with an $m \times p$ matrix over an arbitrary coefficient ring using as few multiplications as possible. Notable early milestones in this effort are Laderman's algorithm (Laderman, 1976), which for $(n, m, k) = (3, 3, 3)$ uses 23 multiplications rather than 27, and the paper by Hopcroft and Kerr (1971), which discusses the ranks of all formats of the form $(2, m, p)$.

The minimal number of multiplications required for multiplying an $n \times m$ matrix with an $m \times p$ matrix is known as the *rank* of the matrix multiplication tensor of format (n, m, p) . By slight abuse of language, we also say that the number of multiplications needed by a specific algorithm for carrying out this product is the rank of this algorithm. Note that the rank of an algorithm is an upper bound for the rank of the tensor. While most of the work on matrix multiplication focuses on the tensor rank of asymptotically large matrices, also the ranks for many small matrix formats remain unknown.

Nowadays computers are used for finding for matrix multiplication schemes of low rank. There are several approaches. Smirnov (2013) found several improvements using numerical

¹An earlier version of this work was presented in the poster "Consequences of the Moosbauer-Poole Algorithms" at ISSAC'25 (Kauers and Wood, 2025). M. Kauers was supported by the Austrian FWF grants 10.55776/PAT8258123, 10.55776/I6130, and 10.55776/PAT9952223. I. Wood was supported by the Austrian FWF grants 10.55776/PAT8258123. The article was finalized while the first author was attending the special semester on Complexity and Linear Algebra at the Simons Institute in Berkeley.

Email addresses: manuel.kauers@jku.at (Manuel Kauers), isaac.wood@jku.at (Isaac Wood)

Preprint submitted to Elsevier

October 24, 2025

optimization. Drevet et al. (2011) used a variant of Pan’s trilinear aggregation method (Pan, 1984) to find improvements for various medium-sized formats. Sedoglavic (2017) uses a divide-and-conquer technique to also improve various medium-size formats. Heule et al. (2021, 2019) used SAT solvers to find many new instances matching Laderman’s bound for $(3, 3, 3)$, but found no improvements. AlphaTensor and AlphaEvolve (Fawzi et al., 2022; Novikov et al., 2025) used machine learning to improve the rank bounds for a number of small formats. Kauers and Moosbauer (2023, 2025) introduced flip graphs and found better ranks for $(5, 5, 5)$ by performing random searches in this graph.

Here we continue the development of the flip graph approach. As introduced by Kauers and Moosbauer (2023), the flip graph for a certain fixed format (n, m, k) is a graph whose vertices are all the multiplication algorithms for multiplying an $n \times m$ matrix with an $m \times p$ matrix, regardless of how many multiplications are needed. Two such algorithms are connected by an edge if it is possible to reach one of them from the other by means of a certain transformation. There are three kinds of transformations. The first one, called “flip”, connects two algorithms of the same rank. The second one, called a “reduction”, leads from one algorithm to an algorithm of rank one less. The third transformation, which is called “plus” and was introduced by Arai et al. (2024), leads from one algorithm to an algorithm of rank one higher.

In search for algorithms of low rank, reduction edges are most useful, but they are rare. The flip graph method proceeds by starting from some given algorithm and traversing a random path of flip edges until a node is encountered which has a reduction edge. Then this edge is taken and the search continues. Following a plus edge is counter productive because it increases the rank, but it is still useful to occasionally choose such an edge in order to avoid getting trapped in a part of the graph from which no further reduction edges can be reached.

The technical details of the flip graph search procedure are not important for the present work. What matters is that for every format (n, m, p) , there is one flip graph, and given a node, i.e., an algorithm for multiplying an $n \times m$ matrix with an $m \times p$ matrix, there is a way to search for algorithms for the same format but of smaller rank. In principle, an algorithm of minimal rank can always be found in this way (Kauers and Moosbauer, 2023, Thm. 9), but in practice, the graph is so large that we can easily fail to find an optimal algorithm, and even if we do find one, it cannot easily be recognized that there is no better one.

Whether or not a search succeeds depends partly on luck, partly on the choice of certain parameters (e.g., how often to take a plus edge), partly on the amount of computing power invested, and to a greater extent on the choice of the starting point of the search. Already in their first paper on the subject, Kauers and Moosbauer (2023) noticed that while they were not able to reach a better rank than 97 for $(5, 5, 5)$ when starting from the standard algorithm, they easily reached a rank scheme of rank 95 (over $\mathbb{Z}_2$) when they took as starting point a scheme of rank 96 that had been found by AlphaTensor (Fawzi et al., 2022).

One year later, Arai et al. (2024) found a scheme of rank 94 for $(5, 5, 5)$ (over $\mathbb{Z}_2$) using an incremental approach. Starting from the standard algorithm for $(2, 2, 2)$, they apply a flip graph search to get a scheme of rank 7. Then they extend this scheme to a scheme of rank 11 for $(2, 2, 3)$ by exploiting the block multiplication rule $A(B|C) = (AB|AC)$ for matrices and use it as starting point for another new flip graph search. They proceed in the same way to construct good schemes of formats $(2, 3, 3)$, $(3, 3, 3)$, $(3, 3, 4)$, $(3, 4, 4)$, $(4, 4, 4)$, $(4, 4, 5)$, $(4, 5, 5)$, and finally $(5, 5, 5)$.

The approach of Arai et al. (2024) not only led to improved bounds for $(4, 5, 5)$ and $(5, 5, 5)$, but also required substantially less computation time than the searches which use the standard algorithm as starting point. It thus seems to be a good idea to construct starting points from optimized schemes of neighboring formats. For the present paper, we have explored this approach

further.

We connect the flip graphs for the various formats (n, m, p) to one single graph by adding two new kinds of edges which connect schemes of different formats. The first kind is called *extension*. It connects every multiplication scheme of some format (n, m, p) to a scheme of format $(n, m, p + 1)$ or $(n, m + 1, p)$ or $(n + 1, m, p)$. This is the same operation that was used by Arai et al. (2024).

The second kind of new edges is called *projection*. It connects every multiplication scheme of some format (n, m, p) to a scheme of format $(n, m, p - 1)$ or $(n, m - 1, p)$ or $(n - 1, m, p)$ by simply setting all the variables corresponding to a certain row or column to zero. For example, if we have a $(2, 2, 3)$ multiplication scheme for computing the product

$$\begin{pmatrix} a_{1,1} & a_{1,2} \\ a_{2,1} & a_{2,2} \end{pmatrix} \begin{pmatrix} b_{1,1} & b_{1,2} & b_{1,3} \\ b_{2,1} & b_{2,2} & b_{2,3} \end{pmatrix} = \begin{pmatrix} c_{1,1} & c_{1,2} & c_{1,3} \\ c_{2,1} & c_{2,2} & c_{2,3} \end{pmatrix},$$

then we can obtain a $(2, 2, 2)$ multiplication scheme by setting $b_{1,3}$ and $b_{2,3}$ to zero and dropping the variables $c_{1,3}$ and $c_{2,3}$.

The graph obtained from all the flip graphs by adding all extension and projection edges is called the *meta flip graph*. The purpose of this article is to share a bunch of rank improvements that were obtained by exploring this graph.

2. Searching From Scratch

Given a collection of multiplication schemes of some format (n, m, p) and some rank r , here is what we do to try to reduce the rank. We consider $100nmp$ random paths in the flip graph that start at randomly chosen starting points chosen from the given collection. The length of each path is restricted to $100000nmp$. Most of the edges in the paths are flips, occasionally we also allow plus edges. If a vertex that has a reduction edge, then we do the reduction. As soon as we reach a vertex whose rank is $r - 1$, i.e., as soon as the number of reduction steps taken exceeds the number of plus steps taken, we abort the search on this specific path and record the resulting scheme.

The result of this process is a collection of various schemes of rank $r - 1$, with which we proceed in the same way to obtain a collection of schemes of rank $r - 2$. The process is repeated until we reach a rank at which none of the considered random paths leads to a reduction.

In order to traverse the meta flip graph, where we also allow extensions and projections to change the format of the multiplication algorithms under consideration, we proceed as follows.

We take the standard algorithm for format $(2, 2, 2)$ as starting point, apply the procedure outlined above, and arrive at a collection of schemes of rank 7. From all these schemes, we construct schemes of format $(2, 2, 3)$ by extension. Because of the symmetry of the matrix multiplication tensor, there is no need to also consider the formats $(2, 3, 2)$ or $(3, 2, 2)$. Also, there is no need to consider projections, because formats (n, m, p) with $\min(n, m, p) = 1$ are trivial.

With the schemes of format $(2, 2, 3)$, we apply again the flip graph search procedure to find schemes of smallest possible rank. The resulting collection of schemes is then extended into a collection of schemes of format $(2, 2, 4)$ and a collection of schemes of format $(2, 3, 3)$. At this point, we could in principle also already apply a projection to return to format $(2, 2, 2)$, but we chose to refrain from returning to formats where we have already been.

Continuing in this way, we have explored most of the paths in the grid of formats (n, m, p) of length 11 that start at $(2, 2, 2)$ and don't visit the same format more than once. There are

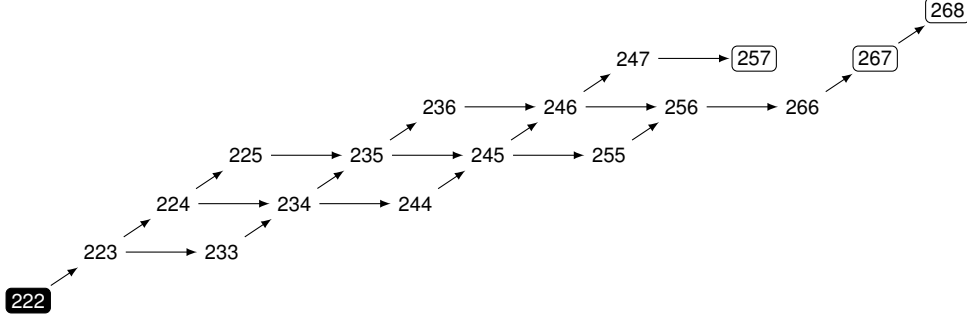


Figure 1: Paths in the meta flip graph starting from the standard algorithm for format $(2, 2, 2)$ (black) that led to improvements for certain formats (boxed).

altogether 1245 such paths, and each format (n, m, p) with $2 \leq \min(n, m, p)$ and $\max(n, m, p) \leq 8$ and $n + m + k \leq 14$ except for $(n, m, p) = (4, 4, 6)$ is visited by at least one of them.

For the formats $(2, 5, 7)$, $(2, 6, 7)$, and $(2, 6, 8)$, we were able to improve the rank bound by one. In the case of $(2, 6, 7)$, there were seven paths that led to the improvement. For each of the two other formats, only one path led to the improvement. Figure 1 shows all the nodes visited by some path that led to an improvement. We also considered paths involving projection steps, but it turned out that none of the paths that led to an improvement contained such an edge.

3. Using other Starting Points

Although we do not know for sure, it is fair to believe that for most of the formats (n, m, p) near $(2, 2, 2)$, the currently best known upper bounds for the rank are actually sharp. This would explain why starting from $(2, 2, 2)$, we have to go some distance before reaching a format where a rank drop can be achieved. However, continuing this approach to even larger format becomes prohibitively expensive as the number of paths grows quickly with their lengths.

An attractive alternative is to take starting points of formats other than $(2, 2, 2)$. Multiplication schemes that were obtained by other techniques are natural choices. For example, Moosbauer and Poole (2025) have recently obtained substantial rank improvements for the formats $(5, 5, 5)$ and $(6, 6, 6)$ by using a variant of the flip graph method that takes symmetries of the matrix multiplication tensor into account. Applying a meta flip graph search as described in the previous section with their schemes as starting points, we were able to obtain rank reductions for a number of formats in the vicinity of $(5, 5, 5)$ and $(6, 6, 6)$, respectively.

Figure 2 shows the formats visited on paths that led to success. These improvements were already announced on our ISSAC’25 poster (Kauers and Wood, 2025). Note that unlike in the previous section, there are now also successful paths containing projection edges. Another difference to the approach from Section 2 is that some improvements are obtained via paths that pass through suboptimal vertices. For example, starting from $(5, 5, 5)$, there are three paths reaching rank 123 for $(4, 6, 7)$ via the vertex $(4, 6, 6)$, although the search only found schemes of rank 106 for this format. The best known rank bound for $(4, 6, 6)$ is 105. Altogether, eight of the 25 paths leading to a rank improvement pass through a vertex for which the best known rank bound was not reached.

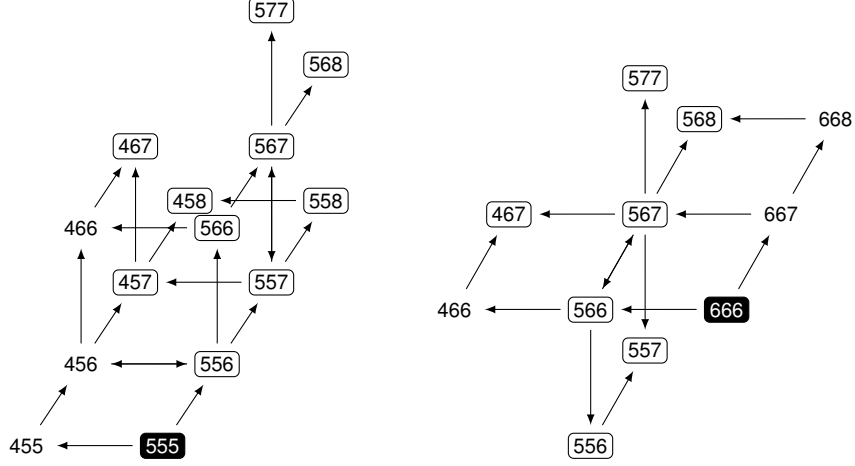


Figure 2: Paths in the meta flip graph starting from a scheme found by Moosbauer and Poole (2025) (black) that led to improvements for certain formats (boxed).

Shortly after the preprint of our ISSAC poster abstract appeared on ArXiv, AlphaEvolve (Novikov et al., 2025) released rank improvements for about a dozen formats. Taking these as starting points and exploring their vicinity in the meta flip graph, we were able to identify some further rank improvements.

4. Coefficient Domains with more than Two Elements

Turning from one algorithm to another algorithm via a flip amounts to adding something and subtracting it at some other position. In order to avoid having to decide what to add and subtract, flip graph searches have so far always been performed with $\mathbb{Z}_2$ as ground domain. This has the advantage that there is only one choice what to add and subtract (because adding and subtracting zero would not change anything). The disadvantage of taking $\mathbb{Z}_2$ as a ground domain is that at the end of the search, we only obtain a multiplication algorithm that is valid for this domain. It would be more interesting to have algorithms with coefficients in $\mathbb{Z}$ or at least in $\mathbb{Q}$, which are applicable to matrices over any coefficient domain in which the denominators of the fractions appearing in the coefficients are invertible.

Hensel lifting (von zur Gathen and Gerhard, 2013) can be used in order to translate an algorithm for the ground domain $\mathbb{Z}_2$ into an algorithm for the ground domain $\mathbb{Z}_{2^\ell}$, for some $\ell \in \mathbb{N}$. Every increment of the exponent ℓ requires solving an inhomogeneous linear system over $\mathbb{Z}_2$. Occasionally, this system happens to be unsolvable, and in this case, no extension to a higher exponent is possible. In most of the cases however, we observe that the algorithms found by the flip graph method with the ground domain $\mathbb{Z}_2$ can be lifted to $\mathbb{Z}_{2^\ell}$ for a sufficiently large ℓ such that rational reconstruction (von zur Gathen and Gerhard, 2013) succeeds turning the algorithm into one for the ground domain $\mathbb{Q}$.

Algorithms with coefficients in $\mathbb{Q}$ that involve one or more fractions with even denominator cannot be found in this way, because such fractions have no valid homomorphic image in $\mathbb{Z}_2$. For this reason, in the computations reported in the previous section we could only explore

the neighborhoods of 9 of the 13 new algorithms found by AlphaEvolve. The other four had coefficients in $\mathbb{Z}[\frac{1}{2}]$ or in $\mathbb{Z}[\frac{1}{2}, i]$. Dumas et al. (2025a,b) succeeded in eliminating i from these schemes, but not $\frac{1}{2}$.

In order to also explore the neighborhoods of the schemes that are not valid mod 2, we implemented the flip graph method for the ground domains $\mathbb{Z}_3$ and $\mathbb{Z}_5$. They both contain multiplicative inverses of 2, and the latter in addition contains a root of the polynomial $x^2 + 1$. Unfortunately, these efforts were in vein. While in some cases we succeeded in matching the currently best known rank bound, we did not encounter any improvement.

5. Another Search from Scratch

In an extension step, one of the three dimensions of a matrix multiplication format (n, m, p) is increased by one. For example, an algorithm for format $(3, 4, 6)$ can be obtained from an algorithm for format $(3, 4, 5)$ by observing that the matrix product

$$\begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} & a_{1,4} \\ a_{2,1} & a_{2,2} & a_{2,3} & a_{2,4} \\ a_{3,1} & a_{3,2} & a_{3,3} & a_{3,4} \end{pmatrix} \begin{pmatrix} b_{1,1} & b_{1,2} & b_{1,3} & b_{1,4} & b_{1,5} & b_{1,6} \\ b_{2,1} & b_{2,2} & b_{2,3} & b_{2,4} & b_{2,5} & b_{2,6} \\ b_{3,1} & b_{3,2} & b_{3,3} & b_{3,4} & b_{3,5} & b_{3,6} \\ b_{4,1} & b_{4,2} & b_{4,3} & b_{4,4} & b_{4,5} & b_{4,6} \end{pmatrix}$$

can be carried out by computing the matrix product

$$\begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} & a_{1,4} \\ a_{2,1} & a_{2,2} & a_{2,3} & a_{2,4} \\ a_{3,1} & a_{3,2} & a_{3,3} & a_{3,4} \end{pmatrix} \begin{pmatrix} b_{1,1} & b_{1,2} & b_{1,3} & b_{1,4} & b_{1,5} \\ b_{2,1} & b_{2,2} & b_{2,3} & b_{2,4} & b_{2,5} \\ b_{3,1} & b_{3,2} & b_{3,3} & b_{3,4} & b_{3,5} \\ b_{4,1} & b_{4,2} & b_{4,3} & b_{4,4} & b_{4,5} \end{pmatrix}$$

and the expressions

$$\begin{aligned} & a_{1,1}b_{1,6} + a_{1,2}b_{2,6} + a_{1,3}b_{3,6} + a_{1,4}b_{4,6} \\ & a_{2,1}b_{1,6} + a_{2,2}b_{2,6} + a_{2,3}b_{3,6} + a_{2,4}b_{4,6} \\ & a_{3,1}b_{1,6} + a_{3,2}b_{2,6} + a_{3,3}b_{3,6} + a_{3,4}b_{4,6} \end{aligned}$$

which require 12 multiplications.

The additional expressions amount to a matrix multiplication of format $(3, 4, 1)$, so the extension can be viewed as obtaining the new $(3, 4, 6)$ scheme by patching together a multiplication scheme of format $(3, 4, 5)$ with one of format $(3, 4, 1)$.

Besides the possible alternatives to obtain $(3, 4, 6)$ from $(3, 3, 6)$ and $(3, 1, 6)$ or from $(2, 4, 6)$ and $(1, 4, 6)$, there are further options: we can also obtain $(3, 4, 6)$ as a combination of $(3, 4, 2)$ and $(3, 4, 4)$, as a combination of two copies of $(3, 4, 3)$, or as a combination of two copies of $(3, 2, 6)$.

For all formats (n, m, p) with $n + m + p \leq 19$, starting with $(2, 2, 2)$, we have performed flip graph searches for starting points that were obtained by combining two smaller formats in all possible ways. Because of the large number of combinations, we did not consider increasing the number of start points further by also using projections, and we also did not keep track of the full paths that led to improvements. In fact, in this approach it is no longer adequate to talk about ‘paths’ because every starting point now has two predecessors rather than one.

One of the formats for which an improvement could be found is $(3, 4, 6)$. In this case, the start points obtained from $(3, 4, 5)$ and $(3, 4, 1)$ have rank 58 and could be pushed down to rank 56 via flip graph searches. In contrast, the starting points obtained from two copies of $(3, 2, 6)$ have rank 60 but could be pushed down to rank 54.

In the meta flip graph as defined earlier, if a scheme has several incoming edges from schemes of other formats, it means that it can be obtained in various different ways from other schemes by extension or projection. It does not mean that it was obtained by combining the schemes from which the incoming edges originate. One way to refine the definition of the meta flip graph such as to accommodate also the combination of schemes as discussed in the present section is to introduce a dummy vertex for each such combination. Merging two schemes S_1, S_2 into a new scheme S_3 is then expressed edges from S_1 and S_2 to the dummy vertex and one edge from that vertex to S_3 .

In Figure 3, the genealogy of the improvement we found for $(3, 6, 8)$ is displayed in this sense. The starting points for the successful flip graph search for this format were obtained by patching together two schemes of format $(3, 3, 8)$. Other combinations were examined as well but didn't lead to an improvement and are therefore not shown in the picture. For the format $(3, 3, 8)$ itself, the flip graph search has reached schemes of best rank for starting points obtained in three different ways: by extensions of schemes of format $(3, 3, 7)$, by combination of a scheme of format $(3, 3, 6)$ with a scheme of format $(3, 3, 2)$ (taking account that these schemes are equivalent to schemes of format $(2, 3, 3)$), and by combination of a scheme of format $(3, 3, 5)$ with a scheme of format $(3, 3, 3)$. All these best-rank schemes together formed the pool of starting points on which the search for $(3, 3, 8)$ was performed.

6. Summary of Results

A summary of our results is given in the table in Figure 4. Each row in this table indicates a matrix multiplication format for which we were able to improve the previously best known upper bound on the rank as indicated in the second column. These numbers are taken from Sedoglavic (2025), who maintains a table with the best-known ranks for all matrix multiplication formats (n, m, p) with $\max(n, m, p) \leq 32$.

The lowest ranks we found for a specific rank are highlighted in bold face. Other numbers refer to ranks that are better than previously known but not as good as the improvements via some other path. An “=” indicates that the previously best known upper bound (second column) was reproduced. A “>” indicates that the search got stuck at a rank larger than the rank in the second column. White space indicates that the format was not covered in the respective search. Combinatorial explosion limits the number of cases that can be reached with a reasonable amount of computation time in approaches that try out all paths.

“S. 2” refers to the search described in Section 2. “Sect. 5” refers to the search described in Section 5. This setting was run twice, thus we give two columns. The columns labeled “AlphaEvolve” and “M/P” (for Moosbauer/Poole) refer to the searches described in Section 3. The column “E” refers to results of various experiments we performed in an early stage of this project. They were partly obtained as described in Section 2 and partly obtained using the schemes of Moosbauer and Poole as starting points, but we are unable to reproduce along which paths these results were found. We include this column nevertheless because it contains some improvements that were not found by any of the other searches.

It is noteworthy that each approach finds at least one improvement that each of the other approaches misses. For the approach of Section 2, it is the improvement for $(2, 6, 8)$ from 76 to

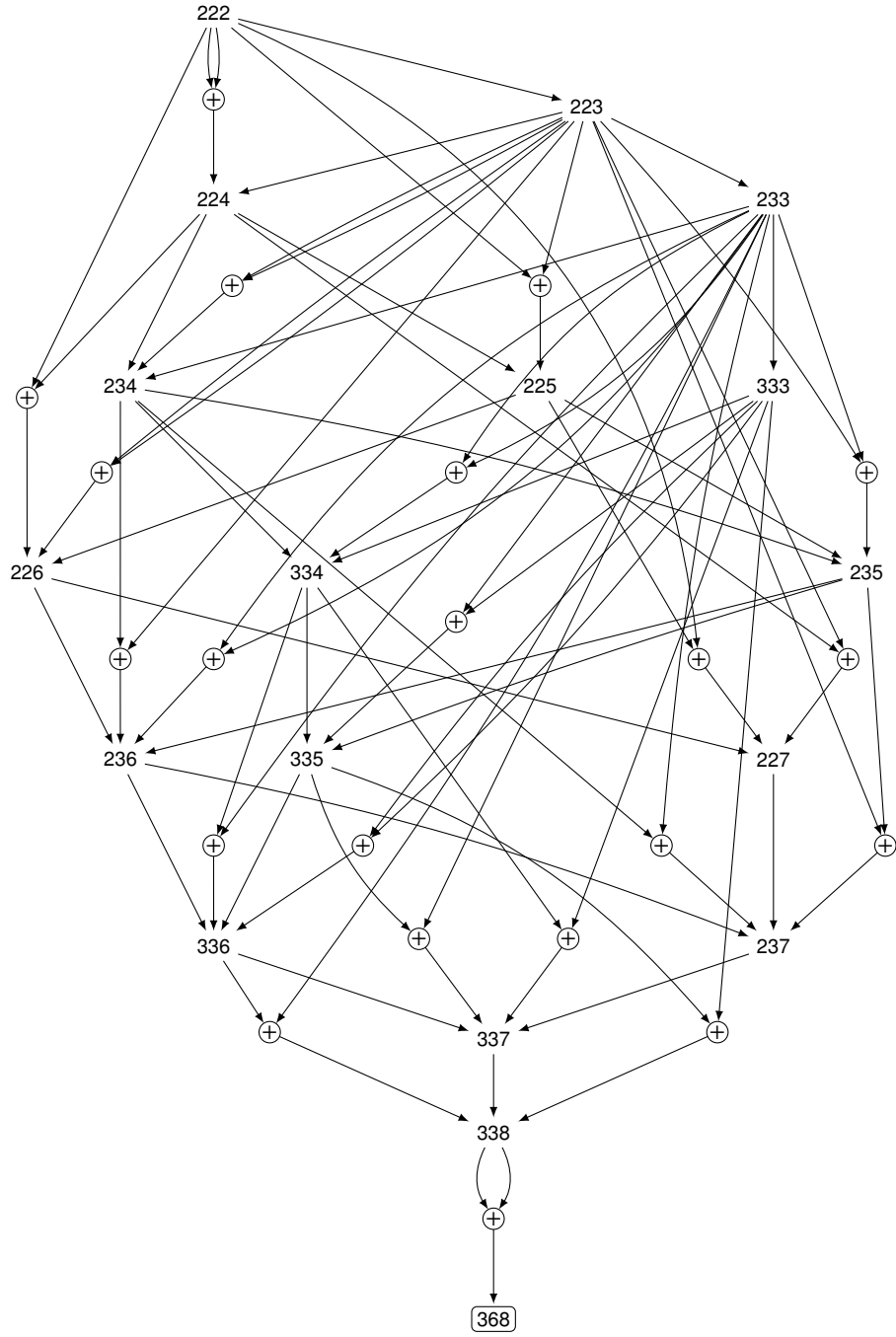


Figure 3: Genealogy of the improvement found for (3, 6, 8).

size	old rank	S. 2	Starting from AlphaEvolve's schemes									M/P		Sect. 5		E
			247	248	256	348	356	357	445	456	447	555	666			
257	56	55	=	=	=	=	=	55		55	=			=	55	55
258	64	>	=	>	=	=	>	>						=	63	
267	67	66	66	>	66	66	=	>						66	66	66
348	74	>	>	>	=	=	>	>	>		>			=	73	
357	80		>	>	>	>	>	=	=	>	>	>		79	=	=
456	90		=		>	>	89	>	>	=	=	>	>	89	89	=
268	76	75	>	>	>	>	>	>						=	>	
277	77	=	>	>	>		>	>						>	>	76
358	93	>	92	>	>	>	>	>		=	92			90	92	
457	109		>	107	>	107	106	107	106	108	104	104		104	104	104
556	116				>		111	115	115	114	112	110	112	111	113	110
377	112		>		>	>	>	>						111	>	>
458	122		>	>	>	>	=	>	=	>	120	118		119	119	
467	125		>		>	>	=	>	=	=	=	123	123	=	>	123
557	133		>		>	>	=	>	=	>	>	127	127	131	131	127
566	137				>		>		>	136		130	130	133	136	130
378	130			>		>	>	>						129	129	
477	147						146	>		145	146			>	>	144
558	151			>		>	>	>		>	>	144		148	146	
567	159					>	>	>		157	>	152	150	158	158	150
388	149					>								145	147	
568	176					>						172	170	>	>	
577	185											177	181	184	>	176
667	185											>	>	>	>	183

Figure 4: Summary of results.

75. For the approach of Section 3, it is for instance the improvement for $(4, 5, 8)$ from 122 to 118. Interestingly however, all the improvements we found starting from one of AlphaEvolve’s schemes was also found in some other way. The approach of Section 5 was most effective.

As explained in Section 4, we were not able to find any improvements running the flip graph method for some ground field other than $\mathbb{Z}_2$. All the results documented in the table were thus first obtained modulo 2. In most of the cases, at least one scheme of smallest rank could be lifted to schemes with coefficients in $\mathbb{Z}$, but not always. For the formats $(4, 5, 6)$, $(3, 7, 8)$ and $(3, 8, 8)$, the improvement appears to work only over $\mathbb{Z}_2$. The other exceptional formats are $(2, 5, 7)$, $(2, 5, 8)$, $(3, 4, 8)$, $(3, 5, 7)$, $(2, 6, 8)$, $(2, 7, 7)$, and $(3, 7, 7)$, where Hensel lifting did not lead to integer coefficients but only to coefficients in $\mathbb{Q}$.

More precisely, for $(2, 5, 8)$, we found a scheme with coefficients in $\mathbb{Z}[\frac{1}{3}, \frac{1}{5}, \frac{1}{7}]$, for $(2, 6, 8)$, our scheme has coefficients in $\mathbb{Z}[\frac{1}{3}, \frac{1}{5}, \frac{1}{11}]$, and for $(3, 4, 8)$, we even need denominators with twelve distinct prime factors. For all other formats where we managed to obtain rational coefficients, these turn out to be in $\mathbb{Z}[\frac{1}{3}]$. For $(3, 7, 7)$, there is in addition one scheme with coefficients in $\mathbb{Z}[\frac{1}{5}]$.

The multiplication schemes announced in this article are publicly available at <https://github.com/mkauers/matrix-multiplication>.

References

- Yamato Arai, Yuma Ichikawa, and Koji Hukushima. Adaptive flip graph algorithm for matrix multiplication. In *Proc. ISSAC’24*, pages 292–298, 2024.
- Charles-Éric Drevet, Md. Nazrul Islam, and Éric Schost. Optimization techniques for small matrix multiplication. *Theor. Comput. Sci.*, 412(22):2219–2236, 2011.
- Jean-Guillaume Dumas, Clément Pernet, and Alexandre Sedoglavic. A non-commutative algorithm for multiplying a 3×4 matrix by a 4×7 matrix using 63 non-complex multiplications. hal-05121550, 2025a.
- Jean-Guillaume Dumas, Clément Pernet, and Alexandre Sedoglavic. A non-commutative algorithm for multiplying 4×4 matrices using 48 non-complex multiplications. Technical Report 2506.13242, ArXiv, 2025b.
- Alhussein Fawzi, Matej Balog, Aja Huang, Thomas Hubert, Bernardino Romera-Paredes, Mohammadamin Barekatain, Alexander Novikov, Francisco J. R. Ruiz, Julian Schrittwieser, Grzegorz Swirszcz, David Silver, Demis Hassabis, and Pushmeet Kohli. Discovering faster matrix multiplication algorithms with reinforcement learning. *Nature*, 610(7930):47–53, 2022. doi: 10.1038/s41586-022-05172-4.
- Marijn J.H. Heule, Manuel Kauers, and Martina Seidl. Local search for fast matrix multiplication. In *Proceedings of SAT’19*, volume 11628 of *LCNS*, pages 155–163, 2019.
- Marijn J.H. Heule, Manuel Kauers, and Martina Seidl. New ways to multiply 3×3 -matrices. *J. Symbolic Comput.*, 104: 899–916, 2021. ISSN 0747-7171. doi: 10.1016/j.jsc.2020.10.003.
- J. E. Hopcroft and L. R. Kerr. On minimizing the number of multiplications necessary for matrix multiplication. *SIAM Journal on Applied Mathematics*, 20(1):30–36, 1971. doi: 10.1137/0120004.
- Manuel Kauers and Jakob Moosbauer. Flip graphs for matrix multiplication. In *Proc. ISSAC’23*, pages 381–388, 2023.
- Manuel Kauers and Jakob Moosbauer. Some new non-commutative matrix multiplication algorithms of size $(n, m, 6)$. *Communications in Computer Algebra*, 58(1):1–11, 2025.
- Manuel Kauers and Isaac Wood. Consequences of the Moosbauer-Poole algorithms. *Communications in Computer Algebra*, 2025. to appear.
- Julian D. Laderman. A noncommutative algorithm for multiplying 3×3 matrices using 23 multiplications. *Bull. Amer. Math. Soc.*, 82(1):126–128, 1976. ISSN 0002-9904. doi: 10.1090/S0002-9904-1976-13988-2.
- Jakob Moosbauer and Michael Poole. Flip graphs with symmetry and new matrix multiplication schemes. In *Proc. ISSAC’25*, 2025. to appear.
- Alexander Novikov, Ngan Vu, Marvin Eisenberger, Emilien Dupont, Po-Sen Huang, Adam Zsolt Wagner, Sergey Shirobokov, Borislav Kozlovskii, Francisco J. R. Ruiz, Abbas Mehrabian, M. Pawan Kumar, Abigail See, Swarat Chaudhuri, George Holland, Alex Davies, Sebastian Nowozin, Pushmeet Kohli, and Matej Balog. Alphaevolve: A coding agent for scientific and algorithmic discovery. Technical Report 2506.13131, ArXiv, 2025.
- Victor Pan. *How to Multiply Matrices Faster*, volume 179 of *LNCS*. Springer, 1984.

- Alexandre Sedoglavic. A non-commutative algorithm for multiplying (7×7) matrices using 250 multiplications. Technical Report 1712.07935, ArXiv, 2017.
- Alexandre Sedoglavic. Yet another catalogue of fast matrix multiplication algorithms. <https://fmm.univ-lille.fr/>, 2025. Accessed: 2025-04-30.
- Alexey V. Smirnov. The bilinear complexity and practical algorithms for matrix multiplication. *Zh. Vychisl. Mat. Mat. Fiz.*, 53(12):1970–1984, 2013. ISSN 0044-4669. doi: 10.1134/S0965542513120129.
- Joachim von zur Gathen and Jürgen Gerhard. *Modern Computer Algebra*. Cambridge University Press, 3 edition, 2013. ISBN 1107039037.