A catalog of fast matrix multiplication algorithms with exhaustive derivations*

Benoit Lacelle
benoit@solven.eu

Draft, July 19, 2026

Abstract

The 2022–2026 burst of activity in small-format matrix multiplication (AlphaTensor 2022, AlphaEvolve 2025, Schwartz–Zwecher 2025, and Perminov's open-source flip-graph framework [18, 17]) has produced striking results but scattered them across fields, attribution conventions, and serialisation formats. We present a unified, machine-checkable catalog covering shapes up to $\langle 32, 32, 32\rangle$ over $\mathbb{Q}$, $\mathbb{Z}$, $\mathbb{R}$, $\mathbb{C}$, $\mathbb{F}_2$, and $\mathbb{F}_3$, with a separate axis for commutative algorithms (Waksman 1970, Makarov 1986, Rosowski 2019). Derivation over the catalog applies a fixed set of well-defined operators exhaustively — axis-flip, Kronecker (with its serendipitous "bud" variant), axis concatenation, recombination-with-allocation (with optional output peeling and leaf-level pair fusion), and downward projection — iterated to a fixed point (Section 4). Our derivation layer is closest in spirit to Sedoglavic's FMM-Lille catalog [24], which likewise layers recursive derivations (Kronecker with serendipity, concatenation, projection) over known bases; we differ by covering more fields ($\mathbb{Z}$, $\mathbb{F}_2$, $\mathbb{F}_3$, and a commutative axis alongside $\mathbb{Q}/\mathbb{R}/\mathbb{C}$) and by exploring the derivation space more exhaustively — rather than sampling it stochastically, as the flip-graph / meta-flip-graph methods [16] do.

As of the 2026-07-12 snapshot over $\mathbb{Q}$, our derivations improve on the best external catalog (FMM-Lille or Perminov) at 908 shapes — 19% of the 4760 comparable shapes in the large-format band $17 \leq \max(n, m, p) \leq 32$; in fact every strict win we produce lands in that band, small formats being saturated. We tie the external best at a further 3688 shapes and trail on only six audited ones, each explained by a single sharing device (fusion kernels) that our composition deliberately forgoes.

We refresh the DIS09 comparison tables, split per field with a commutative column, and provide tooling to regenerate them as the catalog evolves.

1 Introduction

The complexity of multiplying two $n \times n$ matrices has been studied since Strassen's 1969 algorithm [28] broke the cubic barrier, and remains open: the exact value of the matrix multiplication exponent ω is unknown, though sub-cubic upper bounds have been driven down to $\omega < 2.371\ldots$ via the laser method and its successors. The asymptotic regime has been productive for theoretical upper bounds but has produced no concrete algorithms one could implement for finite n; the constants are astronomical.

Meanwhile, the *small-format* regime – exact rank bounds for fixed $\langle n, m, p\rangle$ tensors with $n, m, p \leq 32$ or so – has lived a quieter life dominated by hand-constructed schemes: Strassen (1969) [28], Hopcroft–Kerr (1971) [6], Laderman (1976) [11], Pan (1978, 1980) [14, 15], and Smirnov (1986, 2013) [25, 26]. These results are concrete algorithms; they multiply small matrices using a count of bilinear products that beats the naïve nmp. When applied recursively, each small-format

*Source repository: github.com/solven-eu/matmulcatalog. This article is generated from `paper/article.tex` alongside the catalog; numerical tables track the catalog by construction.

rank improvement compounds into an ω improvement. The strongest *practical* small-format exponent comes from Smirnov's $\langle 3,3,6\rangle$ scheme of rank 40 [26]: at $\omega = 3\log 40/\log 54 \approx 2.7743$ it sits below the exponent of every explicit cubic scheme in the catalog, making it the best base case a recursive implementation can realistically use.

The 2022–2026 period has shaken this landscape, with a fresh record almost every year. In 2022, AlphaTensor [5] found 47 bilinear products for $\mathbb{F}_2\langle 4,4,4\rangle$. In 2024, Kaporin [8] gave an explicit complex 48 for $\mathbb{C}\langle 4,4,4\rangle$ via a semi-analytical solution of the Brent equations; in 2025, AlphaEvolve [13] independently found 48 for the same shape, which Dumas–Pernet–Sedoglavic [4] rationalised the same year — 48 with coefficients in $\{\pm 1, \pm\frac{1}{2}, \pm\frac{1}{4}, \pm\frac{1}{8}\}$, valid over $\mathbb{Q}$ and hence $\mathbb{R}$. Also in 2025, Schwartz–Zwecher [22] produced a systematic disjoint-sum construction giving record ranks at several large shapes. The FMM catalog at Lille [24] and Perminov's `FastMatrixMultiplication` repository [16] both aggregate schemes from across the literature *and* contribute their own results, but via different approaches: FMM layers recursive derivations (Kronecker with serendipity, concatenation, projection) on top of known bases, while Perminov runs flip-graph and meta-flip-graph search to discover fresh low-rank schemes (including ternary-integer ones).

The problem. These results live in incompatible formats (Maple scripts, NumPy archives, ad-hoc JSON), use inconsistent shape conventions (sorted vs unsorted $\langle n,m,p\rangle$, with or without explicit field tags), and carry inconsistent attribution. A $\langle 3,3,3\rangle$ entry sourced from AlphaTensor is sometimes cited as "the AlphaTensor scheme" even though rank 23 was established by Laderman [11] decades earlier – only some of AlphaTensor's per-format ranks were genuine improvements over prior SOTA (its $\mathbb{F}_2\langle 4,4,4\rangle = 47$ among them). The same trap exists in every bulk-import flow we have audited.

We argue that a catalog that is machine-checkable, fully attributed, and uniformly serialised is now both possible and necessary. This paper describes the construction of such a catalog, together with an exhaustive derivation search that automatically discovers recombinations of catalog entries.

Field discipline as load-bearing. A bare $\langle n,m,p\rangle$ is ambiguous: different fields admit different ranks, as the $\langle 4,4,4\rangle$ landscape illustrates — 47 over $\mathbb{F}_2$, 48 over $\mathbb{Q}/\mathbb{R}/\mathbb{C}$, 49 over $\mathbb{Z}$. We keep fields separate and require every rank claim in the catalog (and in this paper) to carry a field tag; Section 2 fixes the conventions. Commutative algorithms (Waksman 1970 [29], Makarov 1986 [12], Rosowski 2019 [20]) form a separate axis: valid for scalar matrix multiplication but not liftable to recursive multiplication over non-commutative rings.

What the catalog catalogs. Each entry encodes either

(a) explicit factor matrices (U,V,W) with field tag, addition count, and lineage record; or

(b) a *derived bound* – a rank given by a closed-form construction (Waksman, Rosowski, Pan trilinear aggregation) that we reconstruct from its lineage record on demand; or

(c) a *cited bound* – our last resort, for the edge cases where we trust a published rank and attribute it to its source but do not (yet) hold its explicit factor matrices.

This three-way split lets us include results we cannot yet materialise as schemes without misrepresenting their status, and lets us regenerate the derived layer as our formulas improve.

Derivation over the catalog. Recursive multiplication over the same catalog is performed by a search that recombines existing entries via well-defined operators (Section 4): axis-flip, Kronecker product (including its serendipitous "bud" variant), axis concatenation, recombination with allocation (with optional output peeling and pair fusion), and downward projection – none

of them a disjoint-sum / Schönhage τ-theorem construction (Section 4.8). The search is iterated until no further rank improvements appear at any catalog shape – a *derivation closure* – and the resulting wins are materialised lazily after the search has converged.

The non-overlap property. An observation that we believe is underdiscussed: our composition operators never *discover* sharing between bilinear products while materialising a scheme. A materialised scheme has exactly the rank the search predicted for it — the sum $\sum_k r_{\text{sub-}k}$ of its sub-schemes' ranks for pure block-substitution (Kronecker, concatenation, recombination), or a smaller value where a *known* identity (pair fusion, serendipity, projection) reduces it, counted before materialisation and never found during it. What never happens is a product turning out, once expanded, to do double duty across outputs and merge with another for a free rank saving. That kind of non-obvious sharing — as in Strassen's seven-product $\mathbb{R}\langle 2,2,2\rangle$ scheme, where a product serves several output cells at once — is exactly what hand-crafted schemes are built around (Laderman, Smirnov, AlphaTensor likewise). We argue that this makes "composed scheme matches hand-crafted scheme's rank" a meaningful co-existence claim, not a re-discovery claim. The same property predicts where we *lose*: on the handful of shapes an external catalog still beats us (six against FMM-Lille), every deficit traces to one cross-slot sharing device – a *fusion kernel* – that our operators forgo by construction, so those losses confirm the boundary rather than contradict it. Section 5 develops both directions.

Outline. Section 2 fixes notation, the field-discipline conventions, and the paper's glossary. Section 3 describes the catalog's serialisation, lineage records, and the explicit/cited/derived split. Section 4 catalogs the derivation operators; Section 5 develops the non-overlap property and its measured boundary (fusion kernels). Sections 6 and 7 present the two theory contributions — recombination multisets, and the constructive realisation of the Hopcroft–Kerr bound. Section 8 describes the exhaustive derivation search, and Section 9 the empirical case for unbalanced splits. Section 10 gives the date-stamped head-to-head against FMM-Lille and Perminov, with an anatomy of the wins and losses. Section 11 closes with open questions, and Section 12 records the human/AI division of labour.

2 Notation, field discipline, and glossary

A bilinear algorithm computing the product of an $n \times m$ matrix A by an $m \times p$ matrix B consists of three factor matrices $U \in K^{nm\times r}$, $V \in K^{mp\times r}$, $W \in K^{np\times r}$ over a field (or ring) K, such that for every A, B:

$$(AB)_{i,j} \;=\; \sum_{k=1}^{r} W_{(i,j),k} \Big(\sum_{a,b} U_{(a,b),k}\, A_{a,b} \Big) \Big(\sum_{a',b'} V_{(a',b'),k}\, B_{a',b'} \Big).$$

The integer r is the *rank* (a.k.a. *multiplication count*) of the algorithm, and the quantity this paper focuses on.

Shape notation. A bare $\langle n, m, p\rangle$ is not sufficient – the rank depends on the algebra. We adopt:

- $K\langle n, m, p\rangle$: non-commutative matmul over the field K. E.g. $\mathbb{Q}\langle 3,3,3\rangle = 23$ (Laderman).
- $K\langle n, m, p\rangle^{\mathrm{c}}$ (superscript-c): commutative matmul over K. E.g. $\mathbb{Q}\langle 3,3,3\rangle^{\mathrm{c}} = 21$ (Rosowski) — commutativity saves two multiplications at the same shape.

The value after $=$ is always the *rank* — the number of scalar multiplications, which is the quantity that compounds into an ω bound under recursion. Addition counts are tracked separately (Section 3).

The fields we track.

$\mathbb{Z}$, $\mathbb{Q}$, $\mathbb{R}$ Characteristic-zero and non-commutative-friendly (they lift to recursive matmul). We keep them *distinct*, first-class fields — each with its own rank claim and attribution, never collapsed into one combined $\mathbb{Z}/\mathbb{Q}/\mathbb{R}$ bucket — and use the inclusion $\mathbb{Z} \subset \mathbb{Q} \subset \mathbb{R}$ only to compute which fields a scheme satisfies. At $\langle 4,4,4\rangle$, for instance, $\mathbb{Z}$ stands at 49 (Strassen^2) while $\mathbb{Q}$ and $\mathbb{R}$ reach 48 (Dumas–Pernet–Sedoglavic [4]) — the same shape, genuinely different ranks. Ternary-integer schemes ($\mathbb{Z}$T, coefficients in $\{-1,0,1\}$) are a first-class sub-class of $\mathbb{Z}$.

$\mathbb{C}$ A complex extension; real-valid schemes work immediately. AlphaEvolve's complex-coefficient $\mathbb{C}\langle 4,4,4\rangle = 48$ lives here.

$\mathbb{F}_2$ Characteristic-2. AlphaTensor's $\mathbb{F}_2\langle 4,4,4\rangle = 47$ lives here. Integer-valid schemes reduce mod 2 to give $\mathbb{F}_2$ schemes; the field stampers grant this systematically (the promotion made $\sim$8000 integer schemes visible under the $\mathbb{F}_2/\mathbb{F}_3$ selectors).

$\mathbb{F}_3$ Characteristic-3 (GF(3)) — "ternary modular", not to be confused with the ternary-*integer* ZT sub-class of $\mathbb{Z}$. We track it so the discipline is genuinely field-agnostic, though coverage is currently thin: a dedicated $\mathbb{F}_3$ search is open (Section 11).

Commutative axis. Independent of the field choice, an algorithm may or may not assume that the entries of A and B commute. Commutative-only schemes (Waksman, Rosowski, Makarov 1986) do *not* lift to recursive matmul over non-commutative rings: they save bilinear products by exploiting the freedom $ab = ba$ that disappears as soon as A and B are themselves matrix blocks, though they remain valid for scalar matmul. Rosowski 2019 (Theorem 5, n odd) reaches $K\langle n,n,n\rangle^{\mathrm{c}} \le n(n+1)(n+2)/2 \approx n^3/2$ — about half the n^3 scalar products, but a constant-factor saving that does not lower ω over non-commutative rings. The catalog flags every commutative scheme with a `"commutative": true` tag, and the comparison tables of Section 10 carry a separate commutative column for each field, because cross-contamination silently produces wrong "wins" against historical non-commutative tables; the canonical example is comparing $\mathbb{R}\langle 3,3,3\rangle^{\mathrm{c}} = 21$ (Rosowski, commutative) to $\mathbb{R}\langle 3,3,3\rangle = 23$ (Laderman, non-commutative) as if they were the same metric.

We require every rank claim in the paper to carry both a field tag and a commutativity tag. The reader can recover the metric from the shape.

2.1 Glossary

The paper uses the following vocabulary consistently; variants from the cited literature ("base", "factorisation") are noted inline.

MM. Matrix multiplication. Throughout this paper, an MM *scheme* for shape $\langle n,m,p\rangle$ is a bilinear algorithm (U,V,W) computing $C = A \cdot B$ for $A \in K^{n\times m}, B \in K^{m\times p}, C \in K^{n\times p}$ over a field K.

Atom. A primitive MM scheme not derived from other schemes in our framework — imported from upstream (Perminov, AlphaTensor, AlphaEvolve), produced by a closed-form constructor (Waksman, Rosowski, Pan TA), or declared as such (Strassen, Laderman). Atoms are the leaves of the lineage DAG; some literature calls them *base* schemes, but the same scheme can be a base for one search round and a derived result of another.

Lineage. An expression describing in a constructive, replayable way how to build an MM scheme from atoms via the derivation operators (recombination, Kronecker, concatenation, axis-flip, permutation).

Stub. A scheme persisted as *lineage only* — no explicit (U, V, W) factor matrices on disk — whose concrete matrices are reconstructed on demand by replaying the lineage (Section 3). Used to store large derived schemes compactly and, during search, as the strategy descriptor recorded before a candidate is materialised.

Allocation. One way to partition a product of shape $\langle n, m, p\rangle$ into blocks compatible with an atom of shape $\langle n', m', p'\rangle$: a per-axis composition $(a_1, \ldots, a_{n'})$ with $\sum a_i = n$ on axis A, and analogously on B and C. The atom is applied at the outer level, with sub-products carried by the blocks of the resulting Cartesian-product decomposition.

Shape $\langle n, m, p\rangle$. A matrix-product shape with $A \in K^{n\times m}, B \in K^{m\times p}$, giving $C \in K^{n\times p}$. We always tag the algebra: $K\langle n, m, p\rangle$ for non-commutative, $K\langle n, m, p\rangle^c$ for commutative; the bare $\langle n, m, p\rangle$ form is reserved for contexts where the algebra has just been named.

Optimisation target. This catalog does *not yet* target minimal additions per scheme: two schemes of equal rank at the same shape (Strassen-18 vs Strassen–Winograd-15) induce the same recursion structure and the same ω, so for now we treat addition count as metadata rather than a primary quality measure — though we still record it, and flag specific cases of interest such as the Strassen-18 vs Strassen–Winograd-15 pair above. What we *do* minimise is the multiset of distinct sub-shapes $\langle n', m', p'\rangle$ the recursion induces — fewer distinct shapes means a simpler block decomposition, less catalog dependency, and easier verification. An illustration from the 2026-06 snapshot: our then-best $\langle 17, 17, 17\rangle$=2940 used three distinct sub-shapes where FMM-Lille's then-best 2934 used two — the two-sub-shape recipe is structurally preferred regardless of addition counts. (Both catalogs have since reached 2930 at this shape; the criterion is unchanged.) One caveat on addition figures: the Strassen–Winograd "15" is a *scheduled* count (after common-subexpression sharing), while flat-counting the (U, V, W) forms gives ∼22–24 — so raw addition counts are an unreliable cross-scheme comparison (Section 4.3.1).

Kronecker product. Composition of a scheme at $\langle n_1, m_1, p_1\rangle$ with a scheme at $\langle n_2, m_2, p_2\rangle$ into a scheme at $\langle n_1 n_2, m_1 m_2, p_1 p_2\rangle$ of rank exactly $r_1 \cdot r_2$ (Section 4.2).

Permutation. Rewrite of a scheme under an S_3 permutation of its three axes, yielding a rank-preserving six-element orbit of equivalent shapes (Section 4).

Axis-flip. Rewrite that reverses the row order on any subset of the three axes, yielding a rank-preserving orbit of eight variants per scheme (Section 4.3.2).

Isotropy + coordinate change. The full symmetry group of a bilinear-algorithm tensor: the continuous coordinate changes $GL_n(K) \times GL_m(K) \times GL_p(K)$ together with the discrete S_3 axis symmetry; the isotropy group of a decomposition is the sub-group that fixes it [1]. The *Permutation* and *Axis-flip* rewrites above are the finite *monomial* corner of this group — Permutation is the S_3 relabelling of the three axes, axis-flip a row-reversal (a monomial element of the GL factors) — the rank-preserving, cheaply enumerable symmetries we exploit directly, as opposed to the full continuous group. de Groote's uniqueness theorem for $\langle 2, 2, 2\rangle$ [2] is treated in Section 6.

Commutative vs non-commutative. Defined in the commutative-axis paragraph of this section; every rank claim carries the tag.

3 Catalog architecture

The catalog's source of truth is a tree of *per-scheme* JSON files under `src/main/resources/schemes/section`N`/`, bucketed by the maximum of the three shape dimensions. Each file is one algorithm — one shape,

one field, one commutativity tag. We do not store every scheme the search can build, only the ones worth keeping:

- the *elected* best derived scheme for each shape — we aim for at least one per shape, the current best-in-scope construction;
- *atoms* worth cataloguing in their own right — a low rank, a low addition count, historical provenance, or any other reason to preserve the specimen;
- schemes kept for their *ingredient* value: a serendipity or projection scheme may not itself hold the best rank at its shape yet be actively consumed as a sub-product by another shape's best scheme, so we retain it.

On top of this tree sit a few *aggregated* JSON files under `docs/`, *generated* from it: the manifest `docs/catalog.json` (one row per elected scheme, consumed by the web browser and the comparison tooling), alongside `docs/cited-bounds.json` and `docs/derived-from-cited-bounds.json`.

3.1 Three layers

Explicit schemes. JSON files that carry the full (U, V, W) factor matrices; every explicit scheme passes a randomised spot-check at load time. The filename is a pure cosmetic label (shape, rank, source or derivation note, short content hash): every property — field, addition count, commutativity — is read from the JSON content, never parsed from the name. The hash, the first seven hex digits of a content hash of the factor matrices, gives each scheme a stable identity independent of how it was named upstream.

Derived bounds. A separate JSON file `docs/derived-from-cited-bounds.json` records rank claims that our own constructors can re-derive on demand from formulas (Waksman 1970, Rosowski 2019 Theorem 2/3, Pan 1980 trilinear aggregation, Hopcroft–Kerr 1971 $\langle 2, b, c\rangle$). Generated by `GenerateDerivedBounds.java`; rebuilds whenever the underlying formula changes. This is the layer that grows as we "move" bounds from the cited tier to the derived tier by reproducing the original construction.

Cited bounds. The failover tier. A central JSON file `docs/cited-bounds.json` records rank claims whose explicit factor matrices we do not (yet) hold and which no constructor of ours re-derives — so we neither materialise nor re-derive them, but still trust and attribute the published rank. Each entry has shape, field, commutativity, rank, addition count (or `null`), source citation, and discovery status. This lets us show the result in comparison tables without pretending to have the scheme. The cited layer is generated by `GenerateCitedBounds.java`.

3.2 Lineage records

The *lineage* of a scheme is the exact sequence of composition steps that produced it, and it is **replayable**: a `LineageReplayer` reconstructs the full concrete (U, V, W) factor matrices from the lineage alone. This is essential — it lets the catalog store a large scheme as a lineage-only *stub* (no factor matrices on disk) and rebuild them on demand, and it makes every composed scheme reproducible from its building blocks rather than trusted as opaque numbers. The lineage is a DAG whose leaf nodes are atoms and whose internal nodes are the derivation operators of Section 4, one node type per operator. The lineage is serialised twice: as a human-readable function-call form (e.g. `ConcatRight(KronProduct(strassen, winograd), SAME0)`) and as a machine-readable JSON DAG with shared-subtree references.

The lineage is load-bearing for three reasons:

1. **Independent audit.** Because replay is independent of any stored matrices, re-deriving from leaves and comparing catches silent corruption of an on-disk (U, V, W).

2. **Search-without-materialisation.** The exhaustive derivation search (Section 8) writes a strategy descriptor – effectively a lineage stub – before deciding whether to materialise. This lets the search iterate without paying for materialisation.

3. **Attribution archaeology.** When a bulk-imported catalog re-encodes a scheme originally due to an earlier author, the lineage records the importing source while a separate attribution field (`attribution_for_rank`) records the earliest source that established the bound. Comparison tables pull from the latter.

3.3 Verification

Four verification modes run independently of the search loop.

Spot-check. `Verifier.passesRandomMatmulSpotCheck` samples random A, B, evaluates the scheme via its (U, V, W), and compares against the direct product — $O(S\,r\,(nm{+}mp{+}np))$ for S samples. Cheap enough to run at scheme load time, after every materialisation, and as a nightly catalog-wide sweep; probabilistic, but a few samples over a field make a false pass vanishingly unlikely.

Symbolic. `SymbolicVerifier` checks the trilinear identity *exactly* (an algebraic equality, no floating-point epsilon) and additionally rejects any coefficient outside the declared field. Conclusive, but $O(r\,nm\,mp\,np)$ — the full tensor, $O(r\,n^6)$ in the cubic case — so it is reserved for the small-dimension schemes where that is affordable.

Lineage replay. `LineageReplayer` re-constructs the scheme from its lineage record, requiring the same (U, V, W). Catches divergence between the on-disk scheme and the formula it claims to instantiate.

Field widening. `FieldWideningSweep` re-tags a scheme claimed for a wide field down to the strictest field its coefficients inhabit (e.g. an $\mathbb{R}$-tagged scheme with all coefficients in $\mathbb{Q}$ is really $\mathbb{Q}$). Within characteristic 0 the coefficients *guarantee* the narrower tag — matmul-validity is a system of integer polynomial identities, unchanged by restricting to a subfield — so this is a certain re-tag, not a heuristic.

4 Derivation strategies

A *derivation* strategy takes one or more catalog entries and produces a new bilinear algorithm at a target shape — combining them upward (*composition*: Kronecker, concatenation), spanning a size gap with the mirror pair *decomposition* (tile a large target into blocks handled by a smaller *outer* scheme — a small-shape scheme generating a large one) and its reverse *projection* (restrict a larger scheme down to a sub-shape), or locally reducing the product set (peel, pair fusion); the strategies below are what the search and materialiser currently know how to do. Convention: the *outer* scheme is the one whose bilinear structure is reused, its *sub-schemes* (*leaves*) the ones computing each outer bilinear product at a smaller shape.

A unifying view, and its ceiling: smoothing ω across shapes. Almost every strategy below is *compositional*: it synthesises a scheme at a target shape out of schemes at *other* shapes — the sub-blocks of a decomposition, the factors of a Kronecker product, the concatenands of an axis split — shapes that need not be anywhere near the target (Kronecker *multiplies* them, landing far from either factor). Read on the landscape of ω over shapes, these strategies *interpolate* in ω-value: the rank they produce at $\langle n, m, p\rangle$ is an arithmetic combination (sum, product) of the ranks — hence the ω — of the shapes they consume, so the target's ω is a blend of its inputs'. Hence a structural ceiling: the purely additive or multiplicative blends — axis concatenation, the plain Kronecker product, plain block-split decomposition — *cannot manufacture an ω below all*

of their inputs: if every shape they draw on has higher ω, the arithmetic combination does too. Pushing below every input needs a mechanism that rewrites the scheme's own product set rather than gluing fixed sub-schemes together; flip-graph and meta-flip-graph moves [17, 9] are exactly such a mechanism, which is why they reach ranks our compositional closure cannot.

Two strategies escape this ceiling. The serendipitous (bud) product (Section 4.2.1) — a Kronecker *variant*, so it is this variant and not the plain Kronecker product that can help — lands *below* the naïve rank product $r_1 r_2$ by borrowing and correcting across the factors; and *projection* (Section 4.5), more speculatively, can restrict a parent below the blend by eliminating dead products. Both inject a correction outside the sum or product of the consumed ranks. That they *do* drop the rank below any pure blend is not hypothetical: projection and the serendipitous product each root dozens of our SOTA schemes (39 and 29 respectively, Section 10.4), shapes whose best held rank comes from exactly this correction — so at those shapes the *local* ω is measurably lower, and we can point to the case. What stays open is *propagation*, not the local drop: a lowered ω at one shape lowers the asymptotic exponent, or lifts another shape, only if that shape consumes it, and we have no argument that a local correction propagates the way a flip-graph base improvement provably does — a local mechanism of unproven global reach.

Table 1 is the section's map. The rows that do not reduce below $\sum_k r_k$ fall under the non-overlap property of Section 5; the reducing rows apply only *known* identities at predict time, deliberately kept as separate constructors so that pure recombination never *discovers* sharing.

Table 1: Derivation strategies used by the catalog. "Rank $<\sum$?" = whether the operator can produce a scheme with fewer than $\sum_k r_{\text{sub-}k}$ bilinear products. *Predict-time* means the reduction is a *known* identity, counted before materialisation — no operator *discovers* new sharing while materialising (the non-overlap property, Section 5).

Operator	Source	Rank $<\sum$?
Axis-flip orbit	folklore + this paper §4.3.2	no
Kronecker product	Strassen 1969	no
Axis concatenation	folklore	no
Recombination with allocation	Strassen-style block split	no[‡]
Serendipitous product	Smith 2002; Perminov 2026 (buds)	yes (predict-time)
Downward projection	Sedoglavic 2017; margin this paper §4.5	yes (predict-time)
Output peel	DIS09 / Islam 2009 γ_5	no
Pair fusion (Pan TA family)	Pan 1980 / KGP 2026	yes (paired)
Leaf-level pairing	Pan 1980 + this paper §4.7	yes (paired)
Disjoint-sum / τ[†]	Pan 1980 / Schönhage 1981 / SZ 2025	n/a (imported)

[†] Imported as explicit factor matrices, never constructed by our search (Section 4.8).
[‡] Exactly $\sum_k r_{\text{sub-}k}$ by Theorem 1, bar an accidental leaf coincidence (probed to zero, Section 5).

4.1 Axis concatenation (block-additive composition)

If two schemes share two of the three axes, they concatenate along the third: a $\langle n, m, p_1\rangle$ scheme of rank r_1 and a $\langle n, m, p_2\rangle$ scheme of rank r_2 compose into a $\langle n, m, p_1 + p_2\rangle$ scheme of rank $r_1 + r_2$, computing the two halves of the result independently; analogously along n and m. Sometimes called "plain split along a dimension". Trivial but weight-pulling: axis concatenation is the direct (root) derivation behind 48 of the 908 shapes where our derived scheme strictly beats every external catalog over $\mathbb{Q}$ (5%; Section 10.4) — the cheapest route to "rectangular" shapes (one large axis) from cubic catalog entries.

4.2 Kronecker product (multiplicative composition)

Given a scheme A at $\langle n_1, m_1, p_1\rangle$ of rank r_1 and a scheme B at $\langle n_2, m_2, p_2\rangle$ of rank r_2, the Kronecker product $A \otimes B$ is a scheme at $\langle n_1 n_2, m_1 m_2, p_1 p_2\rangle$ of rank $r_1 r_2$ — the classical recursive substitution behind Strassen's $\omega \le \log_2 7$: applied recursively to $\langle 2^k, 2^k, 2^k\rangle$ it gives 7^k bilinear products. The search enumerates 2-fold Kronecker chains over the pool; higher-arity chains appear via repeated composition.

4.2.1 Serendipitous product (bud decomposition)

The serendipitous product is a multiplicative composition that, unlike the Kronecker product, can land *below* the naïve rank product $r_1 r_2$. Introduced by Smith [27] and generalised by Sedoglavic in FMM-Lille, we follow the bud-based formalisation of Perminov [19] (Definitions 2.9–2.12, §2.6). Kauers, Moosbauer and Wood [9] independently formalised the same fusion (their Section 3, "divide less, conquer more"): recursive calls that share an input, or whose output is used in several positions, are treated as a single larger multiplication — our U- and W-buds — yielding $\omega(\langle 6,6,6\rangle) \le 2.8019$. Their construction is a Schönhage-style *generalized asymptotic sum inequality* applied to the structure of a *given* decomposition: for $\langle 6,6,6\rangle$ they analyse the flip-graph-found Moosbauer–Poole rank-153 decomposition to extract its shareable sub-blocks, rather than searching for them directly. Finding that shareable structure is the same objective as our bud-richness selection, but it is extracted per decomposition, not produced by a general constructor; whether a deterministic constructor (a targeted basis change in the spirit of the de Groote solve used for kin-row unification by Schwartz and Zwecher [22]) could replace the analysis is open.

The key object is a *bud*: a pair (or group) of rank-one terms sharing the *same* u vector (or the same v, or the same w) up to scaling, since $u \otimes v \otimes w$ is invariant under $u \mapsto \lambda u,\ w \mapsto \lambda^{-1} w$. Buds are basis-free: the shared vector may be any linear combination of entries, not a single coordinate. Grouping the r terms of a scheme by their buds decomposes it into *elementary matmul tensors*

$$\mathcal{T} \;=\; \sum_i S_i \,\langle N_i, M_i, P_i\rangle, \qquad r \;=\; \sum_i S_i \, N_i M_i P_i,$$

where a U-bud of k terms is an $\langle 1,1,k\rangle$ block, a V-bud a $\langle k,1,1\rangle$, a W-bud a $\langle 1,k,1\rangle$, combined buds give blocks such as $\langle 2,1,2\rangle$, and every term in no bud is a trivial $\langle 1,1,1\rangle$. The serendipitous product with a second scheme $\langle n_2, m_2, p_2\rangle$ applies it to each elementary block independently and — crucially — realises each enlarged block by the *best known* scheme for its format rather than by the naïve product:

$$\mathcal{T} \otimes \langle n_2, m_2, p_2\rangle \;=\; \sum_i S_i \,\langle N_i n_2, M_i m_2, P_i p_2\rangle, \qquad r_s \;=\; \sum_i S_i \cdot R(\langle N_i n_2, M_i m_2, P_i p_2\rangle).$$

A saving over the Kronecker product arises whenever $R(\langle N_i n_2, M_i m_2, P_i p_2\rangle) < N_i M_i P_i \cdot R(\langle n_2, m_2, p_2\rangle)$ for at least one block; for trivial blocks the two coincide, so a bud-free scheme reduces exactly to the Kronecker product.

Worked example: $\langle 4,8,12\rangle : 272$. The Hopcroft–Kerr $\langle 2,3,4\rangle : 20$ scheme has four U-buds among its twenty terms, so it decomposes as $12 \cdot \langle 1,1,1\rangle + 4 \cdot \langle 1,1,2\rangle$ (note $12 + 4 \cdot 2 = 20$). Taking the serendipitous product with $\langle 2,4,2\rangle$ and using $R(\langle 2,4,2\rangle) = 14$ and $R(\langle 2,4,4\rangle) = 26$,

$$\langle 2,3,4\rangle \otimes \langle 2,4,2\rangle \;=\; 12 \cdot \langle 2,4,2\rangle + 4 \cdot \langle 2,4,4\rangle, \qquad r_s = 12 \cdot 14 + 4 \cdot 26 = 272,$$

a scheme for $\langle 4,12,8\rangle$ ($\langle 4,8,12\rangle$ up to axis permutation), below the Kronecker value $20 \cdot 14 = 280$. The FMM catalog's [24] shorthand "$(\langle 2,3,4\rangle{:}20 - 8) \otimes \langle 2,4,2\rangle + 4\,\langle 2,4,4\rangle$" records exactly this: eight of the twenty terms live in buds and become the four $\langle 2,4,4\rangle$ blocks; the other twelve stay $\langle 2,4,2\rangle$.

Minimal example (and why it is not a free lunch). The smallest serendipitous saving is Strassen's — but the product *consumes* $R(\langle 2,2,2\rangle) = 7$, it does not produce it. The base $\langle 1,1,2\rangle$ computes $a\,b_1$ and $a\,b_2$, which share the single A-form a: one U-bud of size 2, i.e. the block decomposition $\langle 1,1,2\rangle = 1 \cdot \langle 1,1,2\rangle$. Its serendipitous product with $\langle 2,2,1\rangle$ (rank $R(\langle 2,2,1\rangle) = 4$) sends that bud block to the enlarged format $\langle 1\cdot 2, 1\cdot 2, 2\cdot 1\rangle = \langle 2,2,2\rangle$, realised by its best known scheme:

$$r_s \;=\; 1 \cdot R(\langle 2,2,2\rangle) \;=\; 7 \;<\; \underbrace{(1\cdot 1\cdot 2)\cdot R(\langle 2,2,1\rangle)}_{\text{Kronecker product}} \;=\; 2\cdot 4 \;=\; 8.$$

The bud fuses the Kronecker product's two independent $\langle 2,2,1\rangle$ blocks (eight products reusing the same A entries — that reuse *is* the bud) into one $\langle 2,2,2\rangle$ block; the saving $7 < 8$ exists only because $\langle 2,2,2\rangle$ is sub-additive — only because Strassen supplies $R(\langle 2,2,2\rangle) = 7$ as the enlarged-block realisation. Buds expose *which* products can merge; a sub-additive enlarged block — an *input* to the product — is what turns the merge into a saving.

Bud-richness, rank, and which schemes to retain. The rank-*minimal* scheme for a format is often the *wrong* base for a serendipitous product. Build $\langle 6,6,6\rangle$ as an inner $\langle 3,3,3\rangle$ (Laderman $R = 23$; Smirnov $R(\langle 3,3,6\rangle) = 40$, note $40 < 2\cdot 23$) times one of two $\langle 2,2,2\rangle$ bases, under either operator:

$\langle 2,2,2\rangle$ base	Kronecker	serendipitous
Strassen, rank 7 — no buds	$7\cdot 23 = 161$	161 (= Kronecker)
naïve, rank 8 — four U-buds	$8\cdot 23 = 184$	$4\cdot 40 =$ **160**

Strassen's u, v, w are pairwise non-proportional, so it has no buds and serendipity buys it nothing over Kronecker. The naïve rank-8 scheme is bud-rich — each A-entry feeds two outputs, four U-buds $4\cdot\langle 1,1,2\rangle$ — and fusing each with the inner $\langle 3,3,3\rangle$ into a $\langle 3,3,6\rangle$ gives $4\cdot 40 = 160$: the best of the four, from the *worse*-rank base. A serendipitous base is chosen for bud-richness, not rank. Since r_s depends only on the bud *multiset* $\{S_i\,\langle N_i, M_i, P_i\rangle\}$, what must be retained per format is the *Pareto frontier* in (rank, bud multiset): a higher-rank scheme is worth keeping iff its richer buds win some serendipitous product the lower-rank scheme cannot; a scheme of no-better rank whose bud multiset is contained in another's is dominated. Keeping only the rank-minimal scheme per format would discard the winning base.

How operations act on buds. Bud structure is not preserved uniformly, which is why it is tracked explicitly. *Rank reduction destroys buds*: it replaces un-mixed products by independent linear combinations, making u, v, w pairwise non-proportional — empirically every rank-minimal small-format scheme we hold is bud-free. *Kronecker preserves and multiplies buds* (a product term is proportional to another iff both factors are), so each bud lifts to the product. *Axis permutation (S_3) permutes the bud type*, count unchanged. *Projection can only shrink buds*, while concatenating a scheme with a copy of itself *creates* U-buds (a twin with the same u).

Aligning buds under isotropy. The bud structure of $\mathcal{T}_1 \otimes \mathcal{T}_2$ depends on how the factors' bud axes line up: a size-a U-bud aligned with a size-b U-bud yields a size-ab U-bud; a crossed pair (U against V) reinforces nothing. A basis change preserves proportionality, so the continuous GL part of the isotropy group leaves the bud partition *invariant*; only the discrete S_3 permutation moves buds between axes. Hence two levers: orienting the bud-provider $\mathcal{T}_1$ by S_3 selects which target axis carries the k-fold enlargement (a U-bud gives $\langle n_2, m_2, kp_2\rangle$, a V-bud $\langle kn_2, m_2, p_2\rangle$, whose R differ), so the search tries all three orientations; and aligning both factors' bud axes maximises the ab reinforcement for reuse as a base.

$\Omega(NMP)$	formats	serendipity best	%
≤ 5	2013	0	0.0
6	1122	6	0.5
7	979	12	1.2
8	654	11	1.7
9	389	9	2.3
10	178	3	1.7
≥ 11	121	0	0.0
all	5456	41	0.75

Table 2: How often the best known scheme for a format is a serendipitous product, bucketed by the smoothness $\Omega(NMP)$. Winning formats have mean $\Omega = 7.78$, against 6.29 for the whole population.

What it is. The serendipitous product is *exact* and *combinatorial*: it reads buds off the base scheme's factor matrices and looks up sub-block ranks — no numerical optimisation, no border rank. Our bud recognizer follows Perminov's §2.6, and the assembled scheme is certified by the exact symbolic verifier.

Smoothness governs the serendipitous hit rate. A serendipitous product needs the target to factor as $\langle n_1, m_1, p_1\rangle \otimes \langle n_2, m_2, p_2\rangle$ with *both* factors non-trivial, and $\langle N, M, P\rangle$ has $d(N)\,d(M)\,d(P)$ such factorisations (per-axis divisor counts). A *smooth* format (many prime factors per side, e.g. $30 = 2 \cdot 3 \cdot 5$) exposes far more base/inner splits than a prime-power format, each an independent chance for a bud-rich base to meet a sub-additive enlarged inner. Writing $\Omega(NMP)$ for the number of prime factors of $N\,M\,P$ with multiplicity, Table 2 reports, over all 5456 non-commutative formats in the catalogue, how often the *best* scheme we hold is serendipitous.

Below $\Omega = 6$ no serendipitous scheme is ever best: the smallest non-trivial base/inner pair already needs $\Omega \geq 6$ (e.g. $\langle 2, 3, 7\rangle \otimes \langle 3, 3, 3\rangle$). Above the floor the win rate tracks smoothness, peaking near $\Omega = 9$; the fall-off at $\Omega \geq 11$ is a coverage artefact (those formats lie beyond our materialisation cap), not a ceiling on the method. This concerns the practical per-format implied exponent, not the asymptotic ω; operationally: serendipitous effort goes to smooth formats first, and prime cubes are direct-construction targets where the method cannot help.

4.3 Decomposition: block-splitting and the sub-problem multiset

Decomposition, like the composition operators above, builds *up* in shape: it generates a larger target from smaller schemes (a better small scheme lifts the target — an upward edge in the impact DAG of Section 8). What differs is only the direction of *construction* — instead of gluing two given schemes, it starts from the target shape and tiles it with a smaller *outer* scheme. Only *projection* (Section 4.5) builds *down*, restricting a larger scheme to a sub-shape. *Decomposition* splits a large $\langle N, M, P\rangle$ product into a grid of blocks computed by a smaller *outer* scheme of shape $\langle n_o, m_o, p_o\rangle$ and rank r_o: cut each target axis into that many parts and let each of the outer scheme's r_o bilinear products compute one block product of the resulting $n_o \times m_o$ by $m_o \times p_o$ grid — a $\langle 2, 2, 2\rangle$ outer scheme turns the target into *seven* sub-problems, Laderman's $\langle 3, 3, 3\rangle$ into twenty-three. Listing those sub-problems by shape, with multiplicity, gives the *sub-problem multiset* of the decomposition; each is computed independently, so the realised rank is just the sum of their ranks, $R(\langle N, M, P\rangle) \leq \sum_{\sigma \in \text{multiset}} R(\sigma)$, and the multiset is the *only* thing that fixes the cost — order and labelling are irrelevant. With all parts equal the sub-problems are identical and the sum collapses to $r_o \cdot R(\text{block})$, exactly the Kronecker product (Section 4.2); the interesting regime is *unbalanced* parts, where the sub-problems differ.

Padding, zero-sparing, and the max. The realised size of each sub-product is where padding enters — the heart of the matter. Split each axis in two with the larger part first, $M = M_1 + M_2$, $N = N_1 + N_2$, $P = P_1 + P_2$. Each outer product is a (sum of A-blocks) $\cdot$ (sum of B-blocks); adding blocks of unequal shape embeds the smaller into the larger envelope with zeros, and *those zeros are never multiplied*: a product is realised at its genuine nonzero overlap — the *max* of the real parts on each output axis, the *min* on the contraction axis. A product that sums across an axis is charged that axis's *large* part; a product using a *single* block stays at the part it touches. Hence the expensive corner $\langle M_1, N_1, P_1\rangle$ is paid by every all-sum product; the cheap all-small corner $\langle M_2, N_2, P_2\rangle$ only by a product single-block on *every* axis. This padding-max — not the rank, not the additions — is what will separate Strassen from Winograd on unbalanced splits (Sections 4.3.1 and 4.3.2); **which template** and **which orientation** are the two knobs recombination with allocation (Section 4.4) searches over.

4.3.1 Strassen, Winograd, and the structural difference

The two canonical rank-7 algorithms for $\mathbb{R}\langle 2,2,2\rangle$ are Strassen 1969 and Winograd's 1971 reformulation. The punchline, stated once: used as a $\langle 2,2,2\rangle$ template over a split, at a fixed orientation they induce the *same* multiset of sub-problems and differ only in additions (Strassen 1969: 18; Winograd 1971: 15); it is the axis-flip orbit of Section 4.3.2, re-orienting the template, that changes the multiplications — and where the two diverge. The seven products of each, sized under zero-sparing, are tabulated in Appendix A.

The generic multiset. Collecting the realised sizes of either scheme's seven products gives the same multiset:

$$2\,\langle M_1, N_1, P_1\rangle + \langle M_2, N_1, P_1\rangle + \langle M_1, N_1, P_2\rangle + \langle M_1, N_2, P_1\rangle + \langle M_1, N_2, P_2\rangle + \langle M_2, N_2, P_1\rangle.$$

The largest corner $\langle M_1, N_1, P_1\rangle$ appears twice, the all-small corner $\langle M_2, N_2, P_2\rangle$ not at all, and the split is *not* symmetric: the contraction axis N carries its smaller part in three of the seven products, M and P in two each – so one minimises the small-part products by making N the largest axis. With equal parts (even dimensions) the multiset collapses to $7\,\langle M/2, N/2, P/2\rangle$, the textbook recursion.

Worked example: $\langle 3,3,3\rangle$. Split $3 = 2{+}1$ on every axis ($M_1{=}N_1{=}P_1{=}2$, $M_2{=}N_2{=}P_2{=}1$). Both schemes yield

$$2\,\langle 2,2,2\rangle + \langle 1,2,2\rangle + \langle 2,2,1\rangle + \langle 2,1,2\rangle + \langle 2,1,1\rangle + \langle 1,1,2\rangle,$$

i.e. $2\cdot 7 + 4 + 4 + 4 + 2 + 2 = 30$ leaf multiplications (recursing each $\langle 2,2,2\rangle$ with the same rank-7 scheme) – against $7\cdot 7 = 49$ for the naive pad-to-even $\langle 4,4,4\rangle$ embedding and $3^3 = 27$ schoolbook. The $49 \to 30$ gap is the zero-sparing; the further drop from re-orienting the template is deferred to Section 4.3.2.

Why one needs a *set* of bases. Strassen and Winograd are only two points in the space of rank-7 $\langle 2,2,2\rangle$ schemes, and what matters at an unbalanced split is how many single-block products a scheme has — hence which sub-problem multisets its orbit can reach. The pool therefore carries a *set* of rank-7 $\langle 2,2,2\rangle$ bases inequivalent in multiset behaviour, not just the two textbook ones: we enumerate the rank-7 normal forms (Heun 1994; de Groote's classification) exhaustively, one representative per axis-flip orbit. The same idea applies at any base shape, but the space of rank-r schemes at $\langle 3,3,3\rangle$ and up is far too large to enumerate, so there we fall back on the imported/known schemes plus the axis-flip orbit of each.

4.3.2 Axis-flip orbit

Reversing the row order of any factor matrix (U, V, W) preserves shape and rank; the three independent flips generate an orbit of size 8 per scheme (the elementary abelian $\mathbb{Z}_2^3$; see the glossary). Under uniform allocations the flip is a trivial relabeling and the orbit collapses; under *unbalanced* allocations such as $(9, 8)^3$ the orbit members induce different padding patterns and hence different total ranks, since flipping the outer scheme permutes which block-index of the target receives the padding penalty (Section 4.4). The pool stores one representative per orbit; the search re-enumerates the 8 masks analytically (per-product block-support bitmasks; Section 8) at constant-factor cost.

Why the orbit separates Strassen from Winograd. This is where the two rank-7 templates stop being interchangeable. Strassen's seven products are all two-block sums, so under *any* orbit member each still spans a full part on every axis: Strassen keeps its two full $\langle M_1, N_1, P_1\rangle$ products and can never realise the all-small corner. Winograd's two single-block products ($A_{11}B_{11}$, $A_{12}B_{21}$) *can* be flipped onto the small blocks – one becomes $A_{22}B_{22} = \langle M_2, N_2, P_2\rangle$ – so the best Winograd orbit member drops to a single full product and a strictly smaller multiset whenever the split is unbalanced (the gain vanishes only at equal parts). Concretely, $\langle 17, 17, 17\rangle$ at the $(9, 8)^3$ allocation costs 2930 multiplications for the best Winograd orbit member – its lone all-small $\langle 8, 8, 8\rangle$ product replacing one $\langle 9, 9, 9\rangle$ – against 2940 for Strassen. It is a genuine *multiplication* saving, distinct from Winograd's lower addition count.

4.4 Recombination with allocation

The central composition operator in the catalog: the decomposition of Section 4.3 with freely chosen part sizes. A *small* catalog scheme — the outer *atom*, e.g. Strassen $\langle 2, 2, 2\rangle$, typically 2 or 3 per dimension — multiplies a larger target $\langle n, m, p\rangle$: choose three *allocations* (a partition of n into n_o parts, of m into m_o, of p into p_o); each of the atom's r_o outer products becomes a smaller matmul on the corresponding blocks, looked up in the catalog as a sub-scheme, and the assembled algorithm has rank $\sum_k r_{\text{sub-}k}$. Uniform allocation reduces to the Kronecker product (Section 4.2); non-uniform allocation — e.g. $\langle 17, 17, 17\rangle$ via Strassen at $(9, 8)^3$ — covers the gap left by Kronecker chains, at the price of *padding*: a sub-shape derived from two unequal blocks inherits the larger block's size on the relevant axis.

4.5 Projection margin: a death-rate score

Projection is the downward operator: restrict a larger scheme to a smaller shape and dead-code-eliminate the products that die. A higher-rank parent can thereby project to a *lower*-rank child, with a clean closed form that turns "keep the structurally better scheme, not the lower-rank one" into a computable criterion — the downward analogue of the bud score (Section 4.2.1).

A product survives the restriction iff *each* of its three factors U_k, V_k, W_k still has a nonzero inside the kept sub-block; it dies as soon as *any one* of them vanishes there. Fix an axis, say the first (n); only U (input rows i) and W (output rows i) carry an n-index. Write $U_n(k)$ and $W_n(k)$ for the sets of n-indices column k of U, resp. W, touches. Dropping index i kills product k exactly when *either* its α-form collapses ($U_n(k) = \{i\}$: all of U_k sits in the dropped input row) *or* it writes only the dropped output ($W_n(k) = \{i\}$). The *private rank* at i is the count of such products, $\rho_n(i) = \#\big\{k : U_n(k) = \{i\} \text{ or } W_n(k) = \{i\}\big\}$ (an *or*, not an *and*: a product with $U_n(k) = \{i\}$, $W_n(k) = \{i'\}$ dies under dropping i *or* i'). Each axis uses two of the three matrices — $n \to \{U, W\}$, $m \to \{U, V\}$, $p \to \{V, W\}$ — and the *projection margin* of S (single index, best axis) is

$$\mu(S) = \max_{A \in \{n,m,p\}} \max_i \rho_A(i), \qquad R_{\text{after}} = R_{\text{before}} - \mu(S)$$

exactly as a *survivor count* (an upper bound on the child's rank, tightened below), for projecting out the best single index; dropping δ indices on one axis generalises to $\mu_A^{(\delta)}(S) = \max_{|D|=\delta} \#\{k : \mathrm{supp}_A(k) \subseteq D\}$, with $R_{\text{after}} = R_{\text{before}} - \mu^{(\delta)}$.

Projection therefore minimises $R - \mu$, *not* R: a scheme is worth keeping over a lower-rank rival whenever $R_S - \mu(S) < R_{S'} - \mu(S')$, i.e. on the (rank, projection margin) Pareto frontier. The canonical instance is $\langle 29, 30, 30\rangle$: the rank-best parent, a flat $\langle 30, 30, 30\rangle : 12688$ (Schwartz–Zwecher 2025, ALS-style), has $\mu = 62$ and projects to 12626; the higher-rank *structured* $\langle 30, 30, 30\rangle : 12710$ (Pan trilinear aggregation) has $\mu = 122$ and projects to 12588, matching FMM. Structure localises whole product-slabs to boundary slices (large ρ); a flat decomposition spreads every product across all indices (small ρ), projecting worse despite lower rank.

Survivor count versus true projected rank. The identity $R_{\text{after}} = R_{\text{before}} - \mu$ counts the products that *survive* the restriction; that count is exact, but only an *upper bound* on the child's rank, because restricting can make two survivors *proportional* — and a proportional pair is one rank-one term, not two. The mechanism is a linear form losing a coordinate: suppose two products agree on their α- and γ-forms and differ only in their β-forms, one reading $b_{\cdot,k_1} + b_{\cdot,k_0}$ and the other $b_{\cdot,k_1} - b_{\cdot,k_0}$. Both survive the drop of output column k_0 (each still reads $b_{\cdot,k_1}$), yet *after* the drop both compute $b_{\cdot,k_1}$: the two products coincide and fuse into one, for a -1 that μ never counts. (Over a ternary base "proportional" forces "$\pm$-equal", so the fusion is a cheap sign test.) This is the pair fusion of §4.7, but *induced* by the projection rather than pre-existing: a drop can *manufacture* the coincidence that §5 finds absent in un-projected schemes. The smallest instance is fully explicit: our rank-11 $\mathbb{Q}\langle 2, 2, 3\rangle$ carries two products sharing the input form $A_{0,1} + A_{1,1}$ whose outputs differ only in column $c = 1$ — one feeds $C_{0,0}$ and $C_{0,1}$, the other only $C_{0,0}$. Projecting to $\langle 2, 2, 2\rangle$ by dropping column $c = 1$ erases the distinguishing $C_{0,1}$ entry; both products then write $C_{0,0}$ alone under the same input form and fuse, so nine survivors realise rank eight. The effect stays small but is no toy-shape artefact — $\mathbb{Q}\langle 4, 7, 7\rangle = 144$ restricted to $\langle 4, 6, 7\rangle$ leaves 132 survivors but fuses to 131 — and running the ordinary fusion pass on the restricted scheme recovers it, turning the death-rate score into the true projected rank. It also gives the rank-preserving meta-flip below a quantity it can actually lower: support-unioning flips barely move survivor counts, but they do create proportional pairs.

The opposite extreme: $\mu = 0$ and shared-rank neighbours. Nothing forces a drop to spare a multiplication. When no surviving product is *private* to the dropped slice — every product touches a second kept index on that axis — then $\mu = 0$ and the projection inherits the parent's rank unchanged. The same cascade can do both: our $\mathbb{Q}\langle 9, 17, 19\rangle = 1753$ drops one output column to $\mathbb{Q}\langle 9, 17, 18\rangle = 1600$ ($\mu = 153$, a genuine reduction), yet the next drop, to $\langle 9, 17, 17\rangle$, kills nothing ($\mu = 0$), so 1600 is the best known for *both* shapes. This is not the extra output column being free: it is only $R(\langle 9, 17, 17\rangle) \le R(\langle 9, 17, 18\rangle) = 1600$ holding trivially (a projection never raises rank) while no cheaper dedicated $\langle 9, 17, 17\rangle$ scheme is known — an inherited *upper bound*, not a proven rank, and the shared-rank plateau an artefact of the catalog rather than of the shapes.

A small worked instance, and how to *raise* the margin. Ordering our small catalog by μ, the high-margin schemes are uniformly the *structured* ones (Pan trilinear aggregation) and the flip-graph ones (Perminov), never the flat rank-minimal decompositions: the Perminov $\langle 6, 6, 7\rangle : 183$ has $\mu = 26$ on the p-axis and projects to $\langle 6, 6, 6\rangle$ at $183 - 26 = 157$ (not the catalog optimum 153, but the rank-minimal $\langle 6, 6, 7\rangle$ scheme projects worse). At *small* dimension, however, the flatter low-rank cube still wins projection outright (the rank gap dwarfs the margin gap); the structured-cube advantage only flips the winner once the ranks converge, as at $\langle 30, 30, 30\rangle$ above. Constructing a scheme of the same rank but higher μ is open (parallel to the bud question); a promising route is a rank-preserving meta-flip, searched for the μ-maximal representative of a rank

class. Here μ *selects* among existing schemes. Because projection strictly decreases dimension, a new arrival with $\mu > 0$ propagates as a bounded, acyclic, event-driven downward wave; how the search schedules these waves to a fixpoint is described in Section 8.2.

4.6 Zero-peeling: input- and output-side (DIS09 γ_5)

When a recombination over-allocates an axis and pads the excess with zeros, the padded rows/columns are dead and the affected sub-products can be computed at a smaller shape. Two directions, dual under the matmul tensor's S_3 symmetry: *input-side* (a sub-product reads A only in some rows, or B only in some columns — shrink its input extents) and *output-side* (its W-support lands only in positions peeled away after un-padding — shrink its output extents; the Drevet–Islam–Schost 2009 / Islam 2009 γ_5 reduction). The peel-aware recombination shrinks each sub-product to the *elementwise minimum* of both, and the block-split search enumerates the (allocation, peel) patterns automatically (overshoot bounded per axis). The canonical γ_5 case – $\langle 3,3,3\rangle$ via padded $\langle 4,4,4\rangle$ Strassen, the corner product collapsing $\langle 2,2,2\rangle{=}7 \to \langle 1,2,1\rangle{=}2$ – is closed and tested. Peeling is the second padding-fighter after axis-flip choice (Section 4.3.2); the two are complementary – flip distributes padding across products, peel removes it from a product whose slice was dead anyway – and, like axis-flip, it is *not* a sharing-discovery mechanism (Section 5).

Limit. Peeling alone does not close the odd-cubic gaps such as $\langle 17,17,17\rangle{=}2934$ (FMM-Lille's 2026-06 holding; both catalogs have since reached 2930): there, no standard Strassen product has its support inside the peeled corner, so peeling leaves us at 2940. FMM-Lille's 2934 recipe instead pair-fuses the diagonal $\langle 9,9,9\rangle$ products and substitutes the corner with $\langle 8,8,8\rangle$ – a pair-fusion (Section 4.7) plus outer-template substitution, not a peel.

4.7 Pair fusion (Pan trilinear aggregation)

Pan 1980 [15] observed that two same-shape sub-products arising at distinct positions in a larger scheme can be computed *jointly* via a trilinear aggregation taking $abc + ab + bc + ca$ bilinear products in place of the naïve $2abc$; cyclic pair-fusion identifies pairs of outer products with a compatible block-position pattern and substitutes a fused sub-scheme. This is the first derivation strategy that *does* introduce sharing across outer products – but only in a constrained, pattern-matching way, not as a generic discovery mechanism.

When it pays: a rank-density / ω threshold. Fusing is worthwhile only against the *best* independent computation, i.e. $abc + ab + bc + ca < 2\,R(\langle a,b,c\rangle)$ – not against the naïve $2abc$. Writing the leaf's *rank density* $\rho = R(\langle a,b,c\rangle)/(abc)$, this is $\rho > \frac{1}{2}\big(1 + \frac{ab+bc+ca}{abc}\big) \approx \frac{1}{2}$: pair-fusion helps exactly when the leaf is still *"fat"* – its rank not yet driven below about half the naïve abc. For a cubic leaf $\langle k,k,k\rangle = R$ with implied exponent $\omega_{\text{leaf}} = \log_k R$ (naïve $R = k^3 \Rightarrow \omega = 3$), the same condition reads

$$\omega_{\text{leaf}} \;>\; \omega^\star := \log_k \tfrac{k^3+3k^2}{2} \;=\; 3 - \log_k 2 + \log_k\big(1 + \tfrac{3}{k}\big) \;\approx\; 3 - \log_k 2.$$

Worked examples. $\langle 7,7,7\rangle = 249$ has $\rho = 249/343 = 0.726$ and $\omega_{\text{leaf}} = \log_7 249 = 2.835$, above its threshold $\omega^\star = \log_7 245 = 2.827$: it *fuses* (pair cost $490 < 2 \cdot 249 = 498$, saving 8). By contrast $\langle 8,9,9\rangle = 430$ has $\rho = 430/648 = 0.664$, just below its threshold 0.673: *not* fused (pair cost $873 > 2 \cdot 430 = 860$).

Leaf-level pairing: any two slots of any recombination. Pair fusion originally applied only inside dedicated fused-recombination templates. The device generalises much further, on a simple theoretical ground: the leaves of a recombination are *formally independent* bilinear

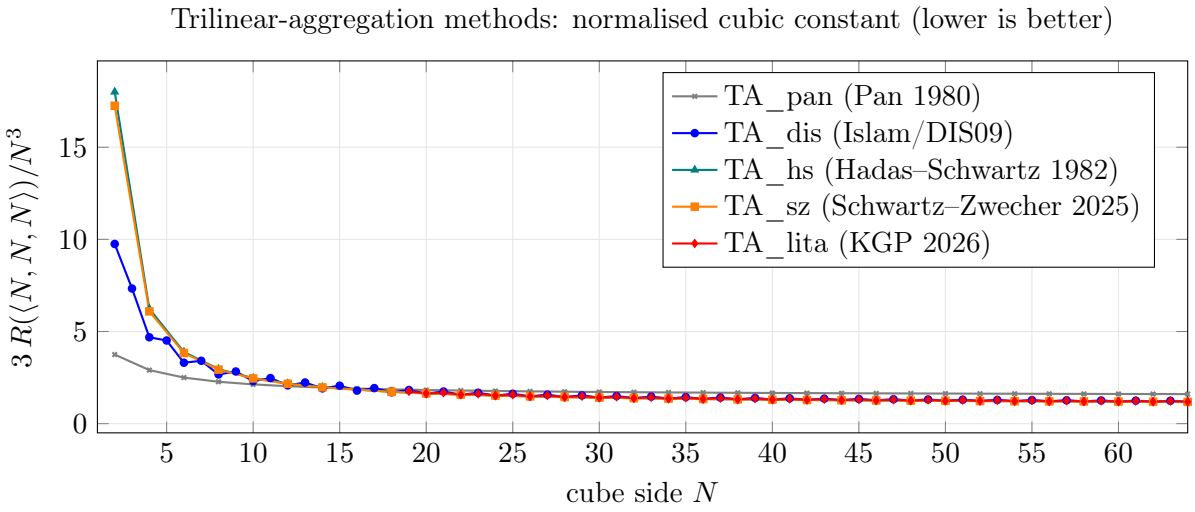

Figure 1: Each trilinear-aggregation method gives an alternative rank for $\langle N, N, N\rangle$ over its own domain (TA_pan/TA_hs/TA_sz are even only, TA_hs/TA_sz exclude the N=16 pole, TA_lita requires $N \geq 19$). Plotted is the normalised leading constant $3R/N^3$; all converge to 1, and the crossovers show which method is tightest over which range — TA_lita (KGP 2026) dominates for odd $N \geq 19$ and large even N, while TA_dis still wins at e.g. N=22.

problems, and bilinear schemes are closed under input substitution — so *any* two same-shape cubic leaves of *any* recombination may be computed jointly at the pair cost $k^3 + 3k^2$, whatever the combining base. (Our engine initially gated pairing to naive-grid outer schemes, on the mistaken belief that a non-trivial combining base could not carry a pair; the block-operand wiring is in fact oblivious to how the slot's inputs are formed.) The search sweeps pairings over every base of small rank, elects a pairing exactly when $k^3 + 3k^2 < 2\,R(\langle k, k, k\rangle)$ at the paired slots, and the lineage records the base, allocations and slot matching (`R*[...]` nodes). Two shapes this closed at exact FMM-Lille ties: $\mathbb{Q}\langle 20, 23, 23\rangle$, $5945 \to 5906$ (two $\langle 11, 11, 11\rangle$ leaves paired at $1694 < 2 \cdot 873$), and $\mathbb{Q}\langle 19, 19, 22\rangle$, $4538 \to 4536$.

This is why pair-fusion is a *small-base, constant-factor* device: as $k \to \infty$ the threshold $\omega^\star \to 3$ while good leaves have ω_{leaf} far below 3, so the saving vanishes asymptotically; the catalog fuses only the two-product instance of Pan's genuinely ω-improving many-product aggregation.

Why pairs, not triples. There is *no* "fuse three *separate* products" primitive: the pair identity fuses two cyclically-related products $\langle a, b, c\rangle + \langle b, c, a\rangle$ at cost $abc + ab + bc + ca$, while the $\frac{1}{3}$ of the full Pan/Islam aggregation comes from one product's own 3-fold cyclic index symmetry, not from fusing three products. A single $\langle n, n, n\rangle$ costs $\approx n^3/3 + O(n^2)$, and the $O(n^2)$ term is itself a *best-of* several Pan-family closed forms: $(n^3 + 12n^2 + 11n)/3$ (Islam 2009 [7] Prop. 1, republished as DIS09 §3) wins for moderate even n; the smaller $\frac{15}{4}n^2$ quadratic of the $\frac{45}{4}n^2$ branch (Hadas–Schwartz 1982, sharpened by Schwartz–Zwecher 2025 [22]) overtakes it for even $n \gtrsim 28$; odd n uses $(n^3 + 15n^2 + 14n - 6)/3$. The newest family member is LITA (*Local Improvements to Trilinear Aggregation*, Khoruzhii–Gelß–Pokutta 2026 [10]): closed forms $R_e(N) = (4N^3+45N^2+116N+84)/12$ for even N and $R_o(N) = (4N^3+57N^2+14N-15)/12-\lfloor 3(N-1)/8\rfloor$ for odd N, valid for $N \geq 19$, which in particular repair the historically weak *odd* case and reproduce several of FMM-Lille's large-cubic index values ($\langle 21, 21, 21\rangle = 5198$, $\langle 23, 23, 23\rangle = 6586$, ...). The chooser takes the *minimum* over all of these branches, so each regime is accounted for, not just the $12n^2$ one. Figure 1 plots the whole family's normalised cubic constant $3R/N^3$ along N: the crossovers show which aggregation is tightest over which range — LITA dominates for odd $N \geq 19$ and large even N, while the Islam/DIS09 branch still wins at, e.g., $N = 22$.

So the real per-leaf choice is shallow pair fusion ($\approx k^3/2$ per product, light $O(k^2)$ corrections)

versus deep single-product TA ($\approx k^3/3$ per product, but $\sim 4\times$ the $O(k^2)$ corrections). At a small leaf the corrections dominate and pairs win decisively. For the seven $\langle 7,7,7\rangle$ leaves of a $\langle 14,14,14\rangle$ (or padded $\langle 17,17,17\rangle$) recombination:

option	per product	total (7 leaves)
solo (catalog SOTA)	249	1743
pair-fused	245	**1719**
full single-product TA	390	2730

Pair fusion (1719) beats both solo (1743) and full TA (2730). The deep TA's $\frac{1}{3}$ only overcomes its heavier corrections at *even* $k \geq 18$; odd k historically carried the heavier $15k^2$ term, until the LITA closed forms [10] repaired the odd case for $k \geq 19$ (Figure 1). This is why the catalog pair-fuses small leaves and reserves full TA for large ones; the chooser picks the cheapest of the three per leaf.

4.8 Disjoint-sum decomposition (τ-theorem)

A target tensor can decompose as a sum of smaller matmul sub-tensors *at the tensor level* – not at a single-axis level – when it admits a τ-theorem-style decomposition: each leg computes a smaller matmul, and the legs combine via the additive structure of the tensor. Schwartz–Zwecher 2025 [22] give explicit constructive disjoint-sum decompositions for *large* formats (e.g. $n = 44, 56, 60$).

We stress that *we do not currently leverage the τ-theorem anywhere.* The disjoint-sum schemes we hold (Schwartz–Zwecher and those FMM/Perminov build this way) are *imported as explicit factor matrices*, not reconstructed from a τ decomposition: no catalog entry carries a τ lineage, and every gap we initially attributed to a τ mechanism turned out to close (or not) by other means – Kronecker, recombination, or a fortunate cousin scheme. A `DisjointSum` lineage node is *reserved* in the schema but exercised by no scheme today; we have no search operator constructing disjoint sums, flagged in Section 11 as the largest open extension.

5 The non-overlap property and a methodological boundary

The composition operators of Sections 4.3.2 through 4.6 are subject to the following structural invariant.

Theorem 1 (Non-overlap)**.** *Let O be an outer bilinear scheme of shape $\langle n_o, m_o, p_o\rangle$ with r_o products, and let $L_1, \ldots, L_{r_o}$ be sub-schemes assigned by block-substitution (the recombination-with-allocation operator, Section 4.4) to a target shape $\langle n, m, p\rangle$ with allocations $(\mathbf{a}_A, \mathbf{a}_B, \mathbf{a}_C)$ and an optional output peel pattern. The resulting algorithm has rank exactly $\sum_k r_{L_k}$, and no two of its bilinear products are equal as bilinear forms.*

Sketch. The construction is block-substitution: each outer product $k \in [1, r_o]$ is instantiated as the sub-scheme L_k wired into block-positions of (A, B, C) determined by the allocations and the outer scheme's bilinear form. Two columns of the resulting (U, V, W) can be equal as bilinear forms only if (a) they come from the same outer product index AND (b) they correspond to the same column of L_k – which would require the same column of the sub-scheme to be wired to two different positions, contradicting the definition of block-substitution. The peel reduction (Section 4.6) removes dead rows of W but never identifies two columns. The axis-flip orbit (Section 4.3.2) is an outer transformation that does not change the rank or the column-distinctness. □

Scope: the operations that *reduce* below the sum. The theorem governs pure block-substitution: Kronecker, concatenation, recombination-with-allocation, axis-flip, peel. Three operations deliberately produce fewer than $\sum_k r_{L_k}$ products, and none is a counterexample, because each applies a *known* reduction at predict time; nothing is discovered during materialisation.

- **Pair fusion (Pan TA)**: an optimisation of a fixed decomposition's product set through Pan's trilinear-aggregation identity (Section 4.7); the saving is a property of that identity, counted before materialisation.
- **The serendipitous product**: replaces an enlarged sub-block by its catalogued best rank — a predict-time lookup.
- **Downward projection**: restricts a materialised parent and runs the same deterministic dead-code elimination that the projection-margin score (Section 8) already predicted.

So the deeper invariant below — *predicted rank equals materialised rank, with no sharing found during materialisation* — holds for these three as well, even though the literal $\sum_k r_{L_k}$ equality of the theorem does not.

Theorem 1 has two practical consequences worth calling out.

Predicted rank equals materialised rank. The exhaustive derivation search (Section 8) propagates rank predictions $\hat{r} = \sum_k r_{\text{sub-}k}$ computed by summing sub-shape lookups in the catalog. The materialised scheme has exactly that rank. There is no hidden discovery during materialisation; the materialisation step is faithful execution of whatever strategy the search picked. This is what lets us split the closure loop into a fast rank-only search phase and a lazy materialise-only phase without losing accuracy – see Section 8 for the algorithm.

A methodological boundary against hand-crafted schemes. Almost every hand-crafted small-format result in the literature exists *because* its author found a non-obvious shared bilinear core: Strassen's $\mathbb{R}\langle 2,2,2\rangle = 7$ beats 8 because two of his seven products do double duty across output cells; Laderman's $\mathbb{R}\langle 3,3,3\rangle = 23$ is the same story at larger scale; Smirnov, Pan, AlphaTensor, AlphaEvolve all hunt for non-obvious cross-product sharing. Theorem 1 says our *composition* operators do *not* hunt for such sharing – the wins come from picking the right composition of cataloged building blocks. (The reducing operators of Section 4.3 ff. — serendipity, pair fusion, projection — do exploit known reductions, but never discover new ones; see the scope remark above.)

This makes a "composition matches hand-crafted rank" outcome a *co-existence* claim, not a re-discovery claim. When the search reached $\mathbb{Q}\langle 17,17,17\rangle = 2930$ (2026-06) via a Winograd recombination at $(9,8)^3$, FMM-Lille's then-best for the same shape stood at 2934, by a route structurally distinct from ours (on inspection, itself a plain recombination — not the disjoint sum its description suggested; see Section 4.8); FMM has since matched 2930. Our scheme is a plain sum of independent sub-products and reproduces none of any hand-crafted construction's shared bilinear identities. The catalog records distinct routes with distinct lineages and distinct `attribution_for_rank`.

The two approaches are complementary, not redundant. Composition expands the SOTA frontier without finding new bilinear identities; hand-crafted constructions find new bilinear identities that can be ingested as new outer schemes, immediately expanding what composition can do.

The boundary, measured. The 2026-07 cross-catalog campaign (Section 10) turned this methodological line into an empirical one, in both directions.

Our schemes contain no accidental sharing. The theorem leaves one loophole open in principle: two *independently built* leaves of a recombination could, by coincidence, feed identical bilinear forms over the target's variables — a coinciding $u \otimes v$ pair would merge into one product (the W-columns add) for a free rank -1. We probed our largest contested scheme ($\mathbb{Q}\langle 17,17,19\rangle = 3267$, 3267 products) for such pairs: *zero* coinciding $u \otimes v$ directions. The invariant holds not only by construction within each substitution but empirically across independent leaves.

What separates us from hand-crafted catalogs is exactly the sharing the theorem forgoes. At campaign end the catalog led or tied FMM-Lille everywhere except six shapes (twelve ranks in total, out of thousands of compared shapes). Every one of the six deficits traces to the same device in the external artifact: a *fusion kernel* — a set of products shared *across* recursion slots. The simplest worked specimen, from FMM's $\langle 8, 27, 30\rangle$ artifact (rank 3736, which we decoded as $\langle 4, 9, 10\rangle{:}250 \otimes \langle 2, 3, 3\rangle{:}15$ with a modified kernel): the plain Kronecker product computes 250 independent "slots", each an instance of $\langle 2, 3, 3\rangle$ costing 15 products. In the artifact, 229 slots are plain — but five *pairs* of slots (s, t) are computed jointly:

$$\underbrace{8}_{\text{plain products of } s} + \underbrace{8}_{\text{plain products of } t} + \underbrace{13}_{\text{mixed products serving both}} = 29 \qquad \text{versus} \qquad 15{+}15 = 30 \text{ independently.}$$

Each mixed product's U-column combines contributions of both slots (its column, reshaped as a base×inner matrix, has rank 2 instead of the rank 1 of every plain Kronecker product — this "Kron-rank fingerprint" is also how the device is detected, since nothing at the lineage or rank-arithmetic level reveals it: the sharing is visible only after expanding the scheme to explicit (U, V, W) and rank-testing each product). One rank saved per pair, five pairs, plus related bookkeeping: rank 3736 where plain composition gives 3750. This is Theorem 1's boundary made concrete — the external catalog's residual edge over our composition closure is *only* this cross-slot sharing device, and conversely no engine of Section 4 can reproduce it without a new, dedicated constructor (Section 11).

The invariant is current-state, not eternal. Theorem 1 depends on the materialiser being pure block-substitution: post-construction column fusion, random sign-salting of sub-blocks, or coefficient mixing between outer products sharing a sub-shape would each let materialised rank drop *below* predicted rank — a feature, but one that breaks the invariant the two-phase search of Section 8 relies on, so any such "salt" must be confined to the final materialisation step or come with re-derived predictions. We decided against salt for now: a pure search-time prediction is easier to reason about and to cache, and avoids subtle correctness bugs at the search/materialise boundary; sharing-style discoveries are deferred to pair fusion (Section 4.7), and the disjoint-sum / τ route (Section 4.8) remains future work.

6 Recombination multisets of a base

Recombining a base at a block decomposition (Section 4.4) sends each of its r products to a smaller multiplication; the resulting *multiset* of r sub-shapes is, for plain additive recombination, a complete rank invariant: the recombined rank is $\sum_k r_{\text{sub-}k}$. The multiset depends only on the support pattern of (U, V, W), not on coefficients, and — with each axis's blocks ordered descending — is *symbolic* in the block sizes (n_i, m_j, p_k); the split $(9, 8)^3 \to \langle 17, 17, 17\rangle$ used below is one instantiation. This section enumerates, by derivation rather than search, which multisets a fixed base can ever produce.

Relation to Sedoglavic's Proposition 1. The enumeration recovers, and strictly generalises, the closed form of Sedoglavic [23], whose Proposition 1,

$$\langle u{+}v, u{+}v, u{+}v\rangle \;\le\; \langle u, u, u\rangle \;+\; 3\,\langle u, u, v\rangle \;+\; 3\,\langle v, v, u\rangle \qquad (u > v),$$

is exactly the recombination multiset of the Strassen $\langle 2, 2, 2\rangle$ base at the two-part allocation $[u, v]$ on every axis (one cube plus two cyclic triples); the headline cases $(u, v) = (4, 3) \to \langle 7, 7, 7\rangle{=}250$ and $(9, 8) \to \langle 17, 17, 17\rangle{=}2940$ are two of its allocations. The enumeration subsumes the formula in three directions: every base, every allocation (unbalanced and $\ge$ 3-part splits included), and — via the dominance frontier of Section 6.3 — the optimal multiset rather than one symbolic recipe.

The excluded case $u = v$ is recovered as Pan trilinear aggregation [14], kept only when cheaper; and the generalisation can beat the proposition at its own shapes: for $\langle 17, 17, 17\rangle$ an allocation in the Strassen orbit dominates the $[9, 8]$ split, $2930 < 2940$. The closed form is thus not a separate construction we wire in, but a single base-and-allocation point of the multiset frontier search.

6.1 The orbit and a decoupling lemma

By de Groote [2] the rank-7 decompositions of $\langle 2, 2, 2\rangle$ form a *single* $\mathrm{GL}_2(\mathbb{Q})^3$ orbit (its discrete isotropy is analysed in [1]). Every scheme is therefore a change of basis of Strassen's, acting on the factors by $U'_k = X^\top U_k Y^\top$, $V'_k = Y^{-\top} V_k Z^\top$, $W'_k = X^{-1} W_k Z^{-1}$ for $(X, Y, Z) \in \mathrm{GL}_2 \times \mathrm{GL}_2 \times \mathrm{GL}_2$. A direct brute-force sweep of ternary schemes is hopeless (≈ 39k candidate rank-one terms per product, $39\mathrm{k}^7$ paths). The structure that rescues it:

Lemma 2 (Per-axis decoupling)**.** *The recombination sub-dimension on a given axis depends on* only one *of X, Y, Z, and only through that matrix's column/row directions. Concretely, on the first axis,*

$$\mathrm{sub}A_k = \mathrm{big} \iff d_0^\top U_k \neq 0 \ \wedge \ \mathrm{perp}(d_1)^\top W_k \neq 0, \qquad X = [\, d_0 \mid d_1 \,],$$

and analogously for the second and third axes via Y and Z — with the caveat that, by the transpose placement of the action, Y and Z act through their row *directions (and the dual tests use adjugate* columns*), so the displayed column formula must not be transplanted verbatim.*

The lemma holds because an invertible factor on the *other* two axes kills no rows or columns of a factor matrix, so the per-axis support is unaffected by it. Consequently the three axes are independent and the set of realisable multisets is the product of three per-axis pattern sets, zipped by product index.

6.2 Exact enumeration

Each per-axis pattern is piecewise-constant in the direction(s) and changes only when a direction meets a null-space of one of the fixed (integer) base factors. The critical directions are therefore finite and rational, so a finite integer-direction sweep enumerates *every* realisable pattern — exactly, with integer arithmetic (adjugates in place of inverses, so zero/non-zero tests are exact). For a two-part axis this is provably complete: the factor null-spaces are the only critical directions, and one generic direction covers the open stratum. For ≥ 3-part axes completeness is certified only *empirically*, by direction-bound stability — a bound, not a proof. We emphasise that this is a *derivation*, not a search over schemes.

Result for $\langle 2, 2, 2\rangle$. Over the full $\mathrm{GL}_2(\mathbb{Q})^3$ orbit there are exactly **40** distinct canonical recombination multisets (canonical = quotiented by the base's S_3 axis-symmetry). The same set covers every rank-7 scheme over any characteristic-0 field: the per-axis strata are cut by *rational* hyperplanes (the null-spaces of the integer Strassen factors), so every stratum nonempty over $\mathbb{R}$ or $\mathbb{C}$ already contains a rational point. The count is stable across direction bounds and identical whether seeded from Strassen or Winograd — an independent check of de Groote's single-orbit theorem. Of the 40, **6** are realised by ternary $\{-1, 0, 1\}$ schemes: those of Strassen, Winograd, AlphaTensor-$\mathbb{Z}$, and Perminov, plus two we register as bases so the search can use them — one matching AlphaTensor-$\mathbb{F}_2$'s support, and a novel base whose multiset is $3 \cdot \langle n_1, m_1, p_1\rangle + \langle n_1, m_2, p_2\rangle + \langle n_2, m_1, p_2\rangle + \langle n_2, m_2, p_1\rangle + \langle n_2, m_2, p_2\rangle$, of independent interest because its triple cubic group admits 3-way trilinear aggregation [14], a cost lever Strassen's own multiset does not expose. The ternary count is a lower bound ("at least 6"): stable under widening the change-of-basis alphabet and the choice of seed, but not certified exhaustive over all ternary schemes. It coincides with the size of the canonical frontier below — suggestive, not proven set-identical.

6.3 Dominance pruning: the frontier

Most canonical multisets can never be rank-optimal and need never be considered. The recombined rank is $\sum_k r_{\text{sub-}k}$, and matrix-multiplication rank is *monotone* — $\text{rank}\langle a,b,c\rangle \le \text{rank}\langle a',b',c'\rangle$ whenever $(a,b,c) \le (a',b',c')$ componentwise (pad the smaller product into the larger). The cost is thus a sum over products of a monotone term, so sub-shape dominance is cost-monotone: if every product of multiset A can be matched to a distinct product of B with A's sub-shape componentwise $\le$ its partner, then $\text{cost}(A) \le \text{cost}(B)$ for *every* base recursion and at *every* allocation (block index 0 = largest block under any split). Only the *dominance frontier* — the non-dominated antichain — can be optimal; pruning to it is lossless for the minimum and allocation/base-independent. This is the exact formalisation of the folklore "$1 \cdot \text{BBB} + 2 \cdot \text{SSS}$ beats $2 \cdot \text{BBB} + 1 \cdot \text{SSS}$": the second is dominated.

Two quotient levels matter, and they differ. The *canonical* frontier quotients by the base's shape automorphisms (dominance is then axis-permutation aware, since rank is); it is exact when the allocation is symmetric or the search also permutes axis assignments. The finer *axis-tagged* frontier applies no quotient and is lossless even for asymmetric allocations, where axis identity is real. Both are computed from the canonical *keys*, not from any seeded representative, so both are seed-independent.

	canonical (mod axis sym.)		axis-tagged	
base	multisets	frontier	multisets	frontier
$\langle 2,2,2\rangle$	40	**6**	144	**12**
$\langle 2,2,3\rangle$	12 217	**21**	23 719	**33**
$\langle 2,3,3\rangle$	62 487	**170**	124 250	**310**

Two observations. First, the frontier is far more robust than the raw enumeration: for $\langle 2,2,3\rangle$ the canonical count still grows with the direction bound ($8\,425 \to 12\,217$) while the frontier holds at 21 across bounds — so the pruned object is stable even where completeness of the full set is only an empirical bound. Second, the frontier cannot collapse to a single point: a product is one corner-type across all three axes at once, so the axes are coupled and several incomparable trade-offs survive (this is exactly why $\langle 17,17,17\rangle$ admits the spread of ranks $2930,\ldots,2958$ from one base).

Operationally this makes selection cheap. For a fixed allocation the cost is *linear in the corner-type count-vector*, $\text{cost} = \sum_{\text{type}} \text{count}[\text{type}] \cdot \text{rank}(\text{type})$; the frontier is allocation-independent, so it is built once and each allocation is scored by one count-vector × rank product over the $\le$ few-hundred frontier members. The frontier thus collapses the multiset axis of the search to a small fixed table, leaving the allocation sweep as the only combinatorially large dimension.

6.4 Generality

The construction is base-agnostic: the same orbit enumeration (`RecombinationMultisetOrbit`) applies to any base $\langle n,m,p\rangle$ over $\text{GL}_n \times \text{GL}_m \times \text{GL}_p$, canonicalised by the base's shape automorphisms (the full S_3 for a cube, one swap for $\langle 2,3,3\rangle$, trivial for an all-distinct shape). It thus answers, for any candidate outer base, *which* sub-shape multisets — hence which trilinear-aggregation and padding-avoidance opportunities — that base can ever expose to the frontier search.

7 A constructive realisation of the Hopcroft–Kerr bound

Hopcroft and Kerr [6] proved $R(\langle 2,p,n\rangle) \le \lceil (3pn + \max(p,n))/2 \rceil$ over any field, by an explicit construction (their Theorem 1). The published proof, however, compresses its hardest steps.

Explicit in the 1971 paper: the three diagonal methods; Lemma 2's three *different*-method pair cases; *one* same-method pair case, $(1, 1,$ bridge-2); the non-constructive existence proofs of Lemmas 1 and 3; and the Case-2 outline with the Z-aggregation identities. Not in the paper: the remaining *five* same-method cases ("the other cases follow by symmetry" — they do not: the natural involutions swap methods 1 and 2 but fix method 3); any concrete augmentation matrix; and the fact that Lemma 3's own sequence forces $(3, 3)$-same-method pairs, a case for which no derivation is given and — as we show below — none exists within the natural product family.

For five decades the bound has effectively been a black box. The community catalogs (FMM-Lille [24], Perminov [16]) credit Hopcroft–Kerr for many $\langle 2, p, n\rangle$ entries, yet at several shapes — as of our 2026-06 full synchronisation of both catalogs — *neither holds a scheme attaining the formula, nor any reference to its value as an attainable bound*: at $\langle 2, 10, 15\rangle$ both publish 234 against the formula's 233; at $\langle 2, 12, 16\rangle$ the best published is 298 against 296; at $\langle 2, 10, 16\rangle$ (formula 248) no attaining scheme is published at all. If a working constructive reading of the 1971 paper existed anywhere, these entries would attain the formula. This section gives a complete, machine-verified constructive realisation, identifies precisely which ingredient resisted (and why), and produces schemes strictly below every published catalog at exactly those shapes.

7.1 The construction, in blocks

Write $Y = \bar{A}X$ with $A \in K^{p\times 2}$ augmented to $\bar{A} = MA \in K^{n\times 2}$, $X \in K^{2\times n}$, and $p \le n \le 2p - 1$. The constructive pipeline decomposes into seven blocks. Blocks (i)–(iii) are Hopcroft–Kerr's (with (iii) only partially explicit in the paper); blocks (iv)–(vii) are this work's, and are what turn the 1971 existence argument into an algorithm.

(i) Diagonal methods. Each internal row i carries a method $c(i) \in \{1, 2, 3\}$ computing y_{ii} with two reusable products: method 1 uses $\{A{=}a_2(x_1{+}x_2)$, $B{=}(a_1{-}a_2)x_1\}$, method 2 $\{C{=}(a_1{-}a_2)x_2$, $D{=}a_1(x_1{+}x_2)\}$, method 3 $\{E{=}a_2x_2$, $F{=}a_1x_1\}$.

(ii) Different-method pairs (HK Lemma 2). A pair (y_{ij}, y_{ji}) with $c(i) \ne c(j)$ costs three new products. A structural fact we use heavily: the (α, β)-method pair emits the *third* method's two products on a signed combination of the rows ($(1, 2) \to E, F$ on $a_i{+}a_j$; $(1, 3) \to C, D$ and $(2, 3) \to A, B$ on $a_j{-}a_i$) — each pair manufactures a *virtual diagonal* for free.

(iii) Same-method pairs via a bridge. When $c(i) = c(j)$, three new products suffice by routing through a third row b, consuming the diagonal and pair products of (i, b) and (b, j). Which bridge methods are derivable is the crux below.

(iv) Bridge selection. In cyclic distance-ordered processing, *any* arc-interior position $b = i + e$ $(1 \le e < d)$ is a legal bridge: both required pairs are already computed. With an alternating $1, 2$-coloring an opposite-method interior position essentially always exists, so the benign cases $(1, 1,$ b2$)$, $(2, 2,$ b1$)$ carry almost all pairs.

(v) Unimodular Lemma-1 matrices: integer schemes. M needs every cyclic p-window nonsingular; we strengthen this to *unimodular* (every window determinant ± 1), which makes the back-substituted W — hence the whole scheme — *integer*, over $\mathbb{Z}$ rather than $\mathbb{Q}$ (the classical Vandermonde rows are numerically unusable here, and dense ± 1 rows can *never* be unimodular for $n - p \ge 2$, every 2×2 ± 1-minor being even). Window determinants of $[I_p; B]$ reduce to three minor families of the augmented block B ($m = n{-}p$ rows): leading, trailing, and sliding m-column minors. A *Euclidean construction* satisfies all three with pure $0/1$ entries and no search: a period-m comb body whose sliding windows are permutation matrices, plus — for $r = p \bmod m > 0$ — a tail of r columns equal to $B(m, r)^\top$. Eliminating the comb's permutation part shows the

crossing and trailing minors of B are exactly the same three families of the tail transposed, so the recursion descends like the gcd and terminates at the pure comb. Every window is still certified by fraction-free BigInteger elimination, and the back-substitution $Ax_j = M^{-1}_{\mathrm{win}(j)}(\bar{A}X)_{\mathrm{win}(j),j}$ is exact integer arithmetic.

(vi) Case 2: circulant matching and the repaired Step 3. For even $p = 2k + 2$, the band of half-width k leaves each column one cell short; the budget — $\lceil (3pn+n)/2 \rceil - n(3k+2)$ — is exactly $n/2$ full pairs at distance $k+1$: a perfect matching in the circulant $i \leftrightarrow i+k+1 \pmod n$, whose $\gcd(n, k+1)$ orbit cycles are matched edge-alternately. Odd cycles each leave one column, completed in pairs by Hopcroft–Kerr's *Z-trick* (their pp. 12–13), which we restate operationally: for leftover columns i_2, i_4 at separation $\delta \le k$, set $i_1 = i_2+k+1$, $i_3 = i_4-k-1$ and treat $\alpha = a_{i_2}+a_{i_3}$, $\beta = a_{i_1}-a_{i_4}$ as virtual rows: their method products already exist as the band pairs' cross-products (by the structural fact in (ii)), so the virtual Lemma-2 cross pair $Z(\beta, \alpha_x), Z(\alpha, \beta_x)$ costs three new products and yields both missing cells after subtracting cells of *complete* columns — including out-of-band cells, recovered exactly as rational combinations through the Lemma-1 window relation. One leftover (n odd) is absorbed by the formula's ceiling.

(vii) Chained augmentation for $n > 2p-1$. One Lemma-1 band covers at most $2p-1$ columns, so for larger n the columns are partitioned into segments of size $s \in [p, 2p-1]$, each built by the band construction and concatenated along the column axis (the products of distinct segments share nothing; ranks add). Because $\max(p, s) = s$ throughout the segment range, per-segment formula values telescope to the global $\lceil n(3p+1)/2 \rceil$ whenever the ceiling slack vanishes: automatic for odd p (each segment cost $s(3p+1)/2$ is an integer), and for even p achievable by using at most one odd-size segment, of parity matching n. Rather than hand-coding these rules, the partition is chosen by a dynamic program over the *achieved* ranks of the segment builds, which also routes around the degraded $g \ge 6$ segment sizes of block (vi) — e.g. p=12, n=36 splits as 16+20 (both exact), not 18+18 (both +1). The chain attains the formula at *every* swept $n > 2p-1$, all parities.

7.2 The bridge-3 cases: a published impossibility, and its operational reversal

The repository's prior analysis proved (Gröbner-verified): *no three rank-1 products complete the* $(2, 2, \text{bridge-3})$ *case over characteristic 0 when the reusable set is* $S = \{C_i, D_i, C_j, D_j, E_b, F_b\}$. That theorem is correct — and turns out not to bind. The emission actually disposes of *twelve* reusables: the three diagonals *and all products of the Lemma-2 pairs* (i, b) *and* (b, j), each independently weightable in the output combination. Over this true set, exact search (120 candidate atoms, rational arithmetic) finds three-product completions; solving for the weights gives the explicit identities

$$\begin{aligned}(2, 2, \text{b3}): \quad y_{ij} &= -C_i + D_i + E_b - A(\delta_{ib}) + G_{ib} + A(\delta_{bj}) - E(\sigma) - G^*,\\ y_{ji} &= -C_i + D_i + F_b - B(\delta_{ib}) - G_{ib} + B(\delta_{bj}) - F(\sigma) + G^*,\end{aligned}$$

with $\delta_{ib} = a_b - a_i$, $\delta_{bj} = a_b - a_j$, $\sigma = a_i + a_b - a_j$ (mirrored on the x-side), and the single genuinely new mixed product $G^* = (a_{i,1} + a_{b,2} - a_{j,2})\,(x_{1,b} - x_{1,j} - x_{2,i})$; the three *new* multiplications are $E(\sigma), F(\sigma), G^*$. An analogous identity closes $(1, 1, \text{b3})$. This vindicates Hopcroft–Kerr's "three additional multiplications" claim for the bridge-3 cases: the published impossibility holds for its narrow S, but the construction was never confined to it.

7.3 The genuine gap, and a geometric limit

The same exact search over the true reusable set finds *no* three-product completion for $(3, 3, \text{bridge-1})$ or $(3, 3, \text{bridge-2})$ — and here the negative can be promoted from a catalog falsification to a

structural theorem. Block-decompose the monomial space by rows $\{a_i, a_b\}$ versus a_j and columns $\{x_i, x_b\}$ versus x_j: the two bridge-sum targets sit, with rank 2, in blocks no reusable product touches, which forces *any* three rank-1 completing atoms (arbitrary linear forms on rows and columns $\{i, b, j\}$ — no candidate catalog) to be supported on all four blocks, pins their coefficient supports, and collapses their spill into the $(\{a_i, a_b\}, \{x_i, x_b\})$ block to two nonzero dyads with factors in the virtual-row space $\langle a_i{+}a_b\rangle$ and virtual-column space $\langle x_i{+}x_b\rangle$, each required to lie in the reusable span. An exact computation shows that intersection is one-dimensional, generated by the *rank-2* virtual diagonal $(a_{i1}{+}a_{b1}) \otimes (x_{1i}{+}x_{1b}) + (a_{i2}{+}a_{b2}) \otimes (x_{2i}{+}x_{2b})$ — it contains no rank-1 element, so no three-product completion exists, for either bridge method (scope: the emitter's nine-product reusable set and atoms local to $\{i, b, j\}$). This is the robust obstruction: Hopcroft–Kerr's own Lemma-3 method sequence places method-3 rows at spacing 2, forcing $(3, 3)$ same-method band pairs — so *the paper's Theorem-1 proof, as written, rests on an underivable case*. Our construction avoids it by keeping method-3 rows pairwise more than k apart, which is possible except at a sharply characterised family: for $n = 3(k{+}1)$ the circulant decomposes into $k{+}1$ triangles, demanding $\lfloor (k{+}1)/2 \rfloor$ Z-pairs; three method-3 rows pairwise $>k$ apart on $C_{3(k+1)}$ would need their arcs to sum to at least $3k{+}6 > n$ — impossible. Those shapes ($g \geq 6$ odd cycles; six in the full sweep range, $\langle 2, 12, 18\rangle$ through $\langle 2, 24, 30\rangle$) land at formula+1 to +3; everything else closes exactly.

7.4 Results

Every scheme is machine-verified at emission by a 20,000-sample randomized spot check, the full residual over every tensor cell, and an *exact-rational symbolic* verification (the bilinear identity $\sum_k U_{ak} V_{bk} W_{ck} = T_{abc}$ checked coefficient-wise over $\mathbb{Q}$, no floating point); beyond that gate, every published file was re-verified by an *independent* checker sharing no code with the generator: 465/465 verify the bilinear identity exactly over $\mathbb{Q}$.

Family	swept	at formula	verified
square $\langle 2, n, n\rangle$	$n \leq 16$	14/14	14/14
odd p, $p \leq n \leq 2p{-}1$	$p \leq 13$	40/40	40/40
even p, $p \leq n \leq 2p{-}1$	$p \leq 14$	52/54	54/54
chained, $2p \leq n \leq 32$	$3 \leq p \leq 16$	196/196	196/196

Highlights, all exact-formula: $\mathbb{Z}\langle 2, 10, 15\rangle{=}233$ — at construction time (2026-06) *strictly below every published catalog* (FMM-Lille and Perminov both listed 234); $\mathbb{Z}\langle 2, 12, 16\rangle{=}296$ (then-published best 298); and $\mathbb{Z}\langle 2, 10, 16\rangle{=}248$ (then absent from all catalogs). The full sweep $3 \leq p \leq 32$, $p \leq n \leq 32$ — **465 schemes, 459 at the exact formula** (the six exceptions are precisely the $g \geq 6$ family of the geometric limit above, at +1..+3) — is emitted into the catalog's constructed tier with per-scheme provenance and is regenerable from the emitter alone; at first (band-range) emission 197 schemes strictly improved on the union of all external catalogs, and the chained range added 22 further strict improvements over everything previously held, our own closure included (e.g. $\mathbb{Z}\langle 2, 15, 32\rangle{=}736$ vs. 741).

Checkpoint: propagation into the community catalogs (2026-07-12). The claims above are date-stamped for a reason — within weeks, both catalogs absorbed parts of this program, illustrating the attribution dynamics of Section 10.3.

- *Perminov* now redistributes 14 of the 465 scheme files, across 11 shapes ($\langle 2, 9, 13\rangle$ through $\langle 2, 13, 14\rangle$, including $\langle 2, 10, 15\rangle{=}233$ and $\langle 2, 12, 16\rangle{=}296$), credited under `schemes/known/matmulcatalog/` — each of them his catalog's best holding for the field. His ternary-integer 233 at $\langle 2, 10, 15\rangle$ is derived downstream of the same scheme. Agreement with Perminov at these shapes is our own work round-tripped, not an independent confirmation.

- *FMM-Lille*'s index at $\langle 2, 10, 15\rangle$ has moved $234 \to 233$, citing Hopcroft–Kerr 1971 — i.e. the closed form. To our knowledge the 465 files remain the only machine-verified witnesses that the formula is attained at these shapes.
- Conversely, for the $g \geq 6$ family our impossibility analysis predicts the formula is *not* attainable with local atoms — and FMM's own artifacts corroborate it: at $\langle 2, 12, 18\rangle$ the index quotes the formula's 333 while the published artifact realises 334 (we verified it locally rigid: no proportional triads, every factor-direction class span-tight), exactly our formula+1; the same index-versus-artifact pattern holds across the family ($\langle 2, 14, 21\rangle$, $\langle 2, 16, 24\rangle$, ...). We reported the discrepancies upstream; whether any unpublished scheme attains the index values there is, at this date, an open question precisely aligned with our impossibility boundary.

7.5 Honesty notes

The construction is an upper-bound realisation; Hopcroft–Kerr proved matching lower bounds only for $\langle 2, 2, n\rangle$ and $\langle 2, 3, 3\rangle$. The $(3, 3, \cdot)$ impossibility theorem is exact over all rank-1 atoms *local to the three rows/columns* $\{i, b, j\}$ and scoped to the emitter's nine-product reusable set; widening the reusable set to all twelve products of both adjacent Lemma-2 pairs (which couples the block decomposition), admitting non-local atoms, or trading a four-product completion against savings elsewhere remain open — these are the only identified routes to the six $g \geq 6$ shapes still at $+1..+3$ over the formula (themselves below every published catalog). Lower bounds beyond Hopcroft–Kerr's own remain out of scope. Short computer-algebra scripts accompanying the catalog reproduce every claim in minutes.

8 Exhaustive derivation search

The exhaustive derivation search converges the catalog to a fixed point under the derivation operators of Section 4. It propagates rank improvements along a dependency DAG of shapes (below) rather than re-deriving everything from scratch; the loop terminates when no shape improves.

8.1 Two-phase structure

A naive closure interleaves *search* (find the best recombination) and *materialise* (build and write the concrete (U, V, W)) at every shape, every round. We split them:

Phase 1 – Search. Iterate rounds. For each shape, run the derivation search to find the best predicted rank $\hat{r} = \sum_k r_{\text{sub-}k}$. Strict improvements over the shape's current best (catalog *or* overlay) are recorded in an in-memory *pending overlay*, keyed by shape. The overlay is consulted by the rank resolver in subsequent rounds, so a win at round N propagates to round $N + 1$ without writing anything to disk.[1] Repeat until a round produces no new overlay entries.

Phase 2 – Materialise. Iterate the overlay in dependency order (smaller maximum-dimension first). For each entry, materialise the winning strategy (`Recombination.constructWithAllocation` or its concat / Kronecker analogue). Sub-product lookups are served from a composed lookup that returns just-materialised winners alongside the on-disk catalog, so a larger winner can use a smaller winner from this same phase as a sub-product. Spot-check the result against the predicted rank, verify, write to disk.

[1] The overlay is also a correctness device: an earlier incarnation wrote each winner to disk and relied on the next round to reload it, but the catalog lookup was loaded once per sweep, so round N+1 silently reused round N's snapshot and never saw the new entries.

Theorem 1 guarantees that the rank propagated by Phase 1 equals the rank produced by Phase 2 for every entry that materialises successfully (the rare failures are infrastructural – e.g. a sub-product that became unavailable between rounds).

The impact DAG, not a full sweep. Shapes and their derivation candidates form a directed acyclic graph of *impacts*: an edge $\sigma \to \tau$ exists whenever a rank improvement at shape σ can lower the predicted rank at τ — because τ Kron-, concat- or recombines a copy of σ (an *upward* edge), or because τ is a projection of σ (a *downward* edge; Section 4.4). A win therefore cannot change any shape that is not downstream of it. The search exploits exactly this: the frontier driver keeps a priority queue of seeds keyed by ω (best first), pops a seed, and on each improvement pushes back *only* the shapes that seed can impact — so it rides the DAG, touching a shape only when one of its sub-shapes has just improved. We do *not* perform a full Cartesian sweep of all shapes against all strategies every round: work is proportional to the impactful edges actually traversed, not to |shapes| × |strategies| × |rounds|. The two cost tiers of this DAG also differ — an upward edge is a cheap rank-arithmetic re-evaluation, whereas a downward (projection) edge requires materialising and DCE-ing the parent — so the closure fans out upward eagerly and schedules the downward work.

Why the split matters. Materialisation involves array allocation, factor-matrix arithmetic, spot-check sampling, and disk I/O: a target at $\langle 17, 17, 17\rangle$ takes a fraction of a second to predict and a few seconds to materialise, and at $\langle 32, 32, 32\rangle$ the spread becomes minutes. Postponing materialisation lets a search round terminate as soon as no new predictions are found, even with dozens of shapes pending materialise work. It also decouples *verification*: full random-matmul spot-checks can run in batch against any materialised scheme later; inside the loop only the cheap predicted-vs-actual rank match is needed to prove the materialise step succeeded.

8.2 Rebuilding the catalog from atoms: why the closure is multipass

The closure's job, stated in the strongest form, is to rebuild the entire catalog from the *atoms* alone (Section 3) by repeatedly applying the operators of Section 4; the round-based fixed point makes this safe without hand-ordering which shape depends on which. For an *upward*-only operator set a fixed point would be overkill: Kronecker, axis concatenation, recombination and the serendipitous product (Section 4.2.1) all build a *larger* shape from *smaller* ones, so a single sweep in increasing maximum dimension suffices — every sub-product is final before its consumer is processed.

The multipass design earns its keep once an operator runs *downward*. The canonical example is *projection*, which derives a scheme for a *smaller* shape by zeroing a slice of a larger scheme and dead-code-eliminating the products that fed only the dropped index; the operator and its *projection-margin* score μ are defined in Section 4.5. A downward edge destroys the acyclicity in dimension: improving $\langle 26, 26, 26\rangle$ can lower $\langle 25, 25, 25\rangle$, which a smaller-first sweep has already finalised, so the catalog must instead iterate to a fixed point, re-projecting neighbours after each cube improvement. This is also the *safer* design — no special-casing of "did some larger shape just improve?"; a projection win is simply absorbed on the next round — and it means cube quality propagates downward through the whole catalog: a single better $\langle n, n, n\rangle$ improves every nearby rectangular and odd-cube shape at once. Operationally the closure fires a bounded *downward wave* on each catalog update: a freshly arrived scheme with $\mu > 0$ is projected into its dimension-neighbours, any neighbour that improves is enqueued for its own projections, and termination is immediate since dimension strictly decreases.

8.3 Pool and allocation enumeration

The derivation search at a shape $\langle n, m, p\rangle$ enumerates, over a pool of outer schemes:

- per-axis allocations of $n/m/p$ into the outer scheme's block counts, optionally filtered by a maximum per-allocation imbalance $\max(\mathbf{a}) - \min(\mathbf{a})$ and capped via a top-K-by-balance pre-selector;
- over-allocation by per-axis padding δ, paired with a corresponding tail-peel pattern (Section 4.6); and
- axis-flip orbit masks of the outer scheme, evaluated analytically via per-product block-support bitmasks, so the 8-mask coverage costs only a constant factor over single-mask evaluation.

The default pool carries a small number of high-value outer schemes (Strassen, Winograd, Smirnov 3-row templates, a few imported Pan / AT / FMM bases); an extended mode adds every leaf catalog entry as an outer template, at which point the capped allocation enumerator becomes load-bearing. Engineering details (caching, pool configuration, budget flags) are in Appendix B.

8.4 Allocation optimisation by branch-and-bound

Enumerating allocations naively is hopeless: for a fixed outer scheme of shape $\langle n_o, m_o, p_o \rangle$, the number of ways to split the target is $\binom{N-1}{n_o-1}\binom{M-1}{m_o-1}\binom{P-1}{p_o-1}$, which blows up well before $N = 32$. Yet the inner problem — given a target, template, and orientation, find the allocation minimising $\sum_\sigma R(\sigma)$ over the sub-problem multiset (Section 4.3) — is solved *exactly* by a branch-and-bound over the allocation grid (`AllocationOptimizer`).

The search fixes the three axis partitions in turn ($\mathbf{a}_N$, then $\mathbf{a}_M$, then $\mathbf{a}_P$), depth-first. Two ingredients make it cheap:

- **A strong incumbent.** The Kronecker (uniform-allocation) rank is a ready upper bound, seeded before the descent; most subtrees then fall to the bound immediately.
- **An admissible lower bound.** At a partial node — some axes resolved, others open — each product is charged the smallest rank it could still take (the resolved axes' sizes, a floor on the open ones). Because R is monotone in the dimensions this never overestimates, so a node whose bound already meets the incumbent is pruned without expansion.

Branch-and-bound returns the *same* optimum as the exhaustive grid while visiting far fewer allocations.

The optimiser runs under a configurable budget — exact, capped by a known upper bound, or an anytime node cap — and every result carries its optimality tier per the discipline of Section 1: an exhaustive descent proves the global minimum *over allocations* for that template and orientation (optimal-within-scope, never the tensor rank). Otherwise the reported rank is a bound unless the run reports exhaustive.

8.5 Sound pruning from cheap candidates

The derivation search has three families of candidate strategies at every target shape: multi-base recombination (heavy — iterates the entire pool, every allocation, every axis-flip mask), *concat-split* (cheap — a single axis split, two SOTA lookups), and *Kronecker* (cheap — nested factor enumeration of n, m, p, all SOTA lookups). Cheap candidates finish in milliseconds; the heavy recombination sweep can take minutes on a single shape at high dim.

We exploit this asymmetry. Concat and Kronecker run *first*, producing an upper bound $\hat{r}_{\text{cheap}} = \min(r_{\text{concat}}, r_{\text{kron}})$. The recombination sweep is then seeded with bestRank $:= \hat{r}_{\text{cheap}}$. Two sound prunes follow:

- *Per-base lower bound.* For an outer base of rank r, the recombination produces r sub-products, each costing ≥ 1; the pair-fusion pass (Section 4.7) only applies pairs whose savings

are positive, so pair-cost is at least the unpaired sum on degenerate-shape sub-products. Hence total $\geq r$ under any allocation. If $r \geq$ bestRank, the entire base is skipped.

- *Best-so-far gate.* After each (allocation, mask) tuple completes, the predicted rank is compared against the running bestRank, updating the candidate only on strict improvement.

We deliberately do *not* prune *inside* the sub-product accumulation loop. A running sum above bestRank does not imply the paired total is above it, since pair-fusion can reduce the sum by tens of percent on cyclic-shape multisets. Adding a slack constant would be a heuristic, and we keep the search at SAT-style soundness: every prune is provably correct.

A canonical case: at $\langle 12, 12, 12\rangle$ the Kronecker pair $\langle 2, 4, 4\rangle$=26 $\otimes$ $\langle 6, 3, 3\rangle$=40 gives 1040 in milliseconds; seeded with that bound, the recombination sweep on the AlphaEvolve $\langle 5, 5, 5\rangle$=93 base (whose best is 1071) drops every non-improving allocation instead of grinding for minutes on worthless intermediate bests. Beyond these provably-correct prunes, any heuristic that drops part of the allocation space (balanced-only filters, top-N caps) is opt-in — a flag the user explicitly enables, never a silent default — because unbalanced cubic targets and bridge-shape (Hopcroft–Kerr type) sub-problems live precisely in the region a balanced-only prune discards.

8.6 Low-ω contamination

A complementary optimisation reshapes the *order* in which shapes are processed during a closure round. The naive iteration is lexicographic over (n, m, p), so the search reaches $\langle 12, 12, 12\rangle$ before it reaches $\langle 17, 17, 17\rangle$ and so on. But the value of an early win depends on its implied ω: a low-ω seed *contaminates* every larger shape that consumes it as a sub-product, silently rewriting those shapes' upper bounds.

We rank candidate outer bases (and known catalog shapes) by their implied exponent $\omega(\langle n, m, p\rangle{:}r) = \log_n(r) \cdot 3/(1 + \log_n(m) + \log_n(p))$ under the convention of Section 2, and process low-ω seeds first: e.g. $\langle 4, 4, 4\rangle$=48 ($\omega \approx 2.7925$; AlphaEvolve over $\mathbb{C}$, the DPS rationalisation [4] over $\mathbb{Q}/\mathbb{R}$), Smirnov $\langle 3, 3, 6\rangle$=40, Pan $\langle 3, 4, 7\rangle$=63, then up the ladder. Targets a seed can contaminate (via Kronecker, recombination, or concat) are re-evaluated in the same round once the seed is on the overlay, and the shape order within each round respects this dependency — every shape that consumes a seed via a known composition rule waits for the seed's search to complete — so a fresh ω reduction propagates without a separate round. The seed selection itself is data-driven, from an automatically regenerated ranking of every pool-eligible base by ω (Appendix B).

9 Balance versus imbalance: the records are unbalanced

A recurring heuristic — explicit in DIS09 and folklore well before — is that recombination should prefer *balanced* block splits and avoid unbalanced ones. We test this directly against the catalog and reach a sharper, two-sided conclusion: **balance is better on average, but the records are unbalanced.** A search that prunes unbalanced shapes is therefore average-case sound yet *record-blind* — it discards exactly the shapes that set the lowest exponents.

Throughout this section the per-shape exponent is $\omega(n, m, p) = 3 \ln R(n, m, p)/\ln(nmp)$, where R is the best catalogued non-commutative rank over $\mathbb{R}$ (catalog snapshot 2026-06; the catalog holds a single $\mathbb{R}$-only scheme, so the same figures hold over $\mathbb{Q}$, the field all later sweeps default to).

9.1 On average, ω rises with imbalance

Table 3 buckets every catalogued shape $\langle n, m, p\rangle$ with $2 \leq n \leq m \leq p \leq 16$ by its imbalance $p - n$, and reports the median ω per bucket.

Table 3: Median ω over all catalogued shapes $2 \le n \le m \le p \le 16$, bucketed by imbalance $p - n$. The median climbs monotonically — a *typical* unbalanced shape is less efficient.

imbalance $p - n$	#shapes	median ω
0	15	2.8155
1	28	2.8148
2	39	2.8177
3	48	2.8180
4	55	2.8169
5	60	2.8193
6	63	2.8214
7	64	2.8229
8	63	2.8274
9	60	2.8299
10	55	2.8332
11	48	2.8402
12	39	2.8536
13	28	2.8616
14	15	2.8695

The median rises from 2.8155 (balanced) to 2.8695 at imbalance 14, staying flat (≈ 2.817) out to imbalance ~ 7 and then climbing steeply — the penalty for imbalance is convex, not linear. This is the average-case fact the avoid-unbalanced heuristic rests on.

9.2 ... but the lowest exponents are unbalanced

The aggregate hides the tail. Among the same shapes, the fifteen smallest ω are almost all unbalanced, and the global minimiser in this range is $\langle 3, 3, 6\rangle$ at $\omega = 2.7743$ — below *every* cubic: $\langle 4, 4, 4\rangle$ (2.7925), $\langle 8, 8, 8\rangle$ (2.7974), $\langle 12, 12, 12\rangle$ (2.7957). The low-ω frontier is dominated by the 2:1-ratio family $\langle 3, 3, 6\rangle$, $\langle 6, 6, 12\rangle$, $\langle 9, 9, 12\rangle$, $\langle 5, 5, 12\rangle$, $\langle 3, 4, 6\rangle$. Pruning unbalanced splits removes precisely these record-holders.

9.3 Why the advantage is not extractable by outer splitting

It is tempting to conclude that a recombination should *seek* unbalanced splits so its sub-blocks land on these efficient shapes. The catalog says otherwise, and the mechanism is instructive. Consider $\langle 9, 9, 9\rangle$ recombined by Strassen $\langle 2, 2, 2\rangle$=7: since $9 = 5 + 4 = 6 + 3$, the balanced split $(5, 4)^3$ and the unbalanced $(6, 3)^3$ both tile exactly (no padding). Table 4 puts their seven-product multisets side by side.

Table 4: Strassen $\langle 2, 2, 2\rangle$ recombination of $\langle 9, 9, 9\rangle$: the seven sub-products grouped by corner tier. The unbalanced $(6, 3)^3$ split saves 63 on the three one-max corners ($\langle 3, 3, 6\rangle$=40 vs 61) but pays +60 on the all-max corner and +12 on the middle tier — net +9, so balance wins here.

corner tier	$(5, 4)^3 \to 504$		$(6, 3)^3 \to 513$		Δ
(multiplicity)	shape	subtotal	shape	subtotal	
all-max ×1	$\langle 5, 5, 5\rangle$=93	93	$\langle 6, 6, 6\rangle$=153	153	+60
two-max ×3	$\langle 5, 5, 4\rangle$=76	228	$\langle 6, 6, 3\rangle$=80	240	+12
one-max ×3	$\langle 5, 4, 4\rangle$=61	183	$\langle 3, 3, 6\rangle$=40	120	−63
total		**504**		**513**	+9

A $\langle 2,2,2\rangle$ split realises seven of the eight corners of $\{a_1,a_2\}\times\{b_1,b_2\}\times\{c_1,c_2\}$, and *one of them is the all-max corner* $(\max,\max,\max)$. Because R grows like size$^{\omega}$ with $\omega\approx 2.8$, pushing the max block from 5 to 6 costs more than the cheap corners recover. The efficiency of $\langle 3,3,6\rangle$ is real, but a single block split cannot isolate it from its complementary all-max corner.

The corollary is constructive: an efficient unbalanced atom is exploited not by *splitting* a target down onto it, but by *composing up* from it via the Kronecker product, where the favourable ratio compounds instead of dragging a complementary max-corner along (cf. 4.8).

9.4 Imbalance can win the split, too

Section 9.3 showed balance winning a single example; it is not the rule. We swept every target $\langle n,m,p\rangle$, $3\le n\le m\le p\le 16$, computing for a fixed base the *exact* rank-minimising allocation by the branch-and-bound of Section 8.4 (balanced incumbent + admissible lower bound, verified against exhaustive search), and flagged every target whose optimal split is strictly cheaper than the balanced split. The search is deliberately *single-base* and *non-recursive*: sub-blocks are costed by the catalog's best rank as a black box, so a "win" means the unbalanced split beats its own balanced sibling for that base — not that it beats the global record.

Unbalanced splits win for 15 targets under Strassen $\langle 2,2,2\rangle$ and 120 under Strassen–Winograd (whose asymmetric product supports reward imbalance far more often). The effect is not confined to rectangular targets: the cubic $\langle 13,13,13\rangle$ is itself a win. Table 5 lists representatives.

Table 5: Targets where the rank-optimal single-base split is *unbalanced* (strictly cheaper than the balanced split). Single base, no recursion; sub-blocks costed by catalog SOTA. Savings are small but the existence is the point: a hard "always split balanced" rule is provably suboptimal.

base	target	balanced	optimal split	best
Strassen	$\langle 13,13,13\rangle$	1434	$(5,8)(6,7)(6,7)$	1432
Strassen	$\langle 6,6,10\rangle$	252	$(3,3)(3,3)(4,6)$	247
Strassen	$\langle 12,12,14\rangle$	1281	$(6,6)(6,6)(6,8)$	1271
Winograd	$\langle 9,13,15\rangle$	1156	$(5,4)(7,6)(7,8)$	1145
Winograd	$\langle 9,14,15\rangle$	1237	$(4,5)(7,7)(8,7)$	1226
$\langle 2,2,3\rangle$	$\langle 5,5,12\rangle$	235	$(2,3)(2,3)(6,5,\mathbf{1})$	232

The last row is the most imbalanced win we found in the sweep ($3\le n\le m\le p\le 16$): for $\langle 5,5,12\rangle$ under base $\langle 2,2,3\rangle$ the optimal split of the size-12 axis is $(6,5,\mathbf{1})$ — it *peels a width-1 strip*. The products that touch the unit block compute $\langle\cdot,\cdot,1\rangle$ sub-shapes at naive-product rank (here $R=4,6,6,9$), costing almost nothing, while the remaining products route through the cheap $\langle 3,3,6\rangle$=40 corner — the unbalanced-extraction mechanism in its purest form.

So the practical rule is not "prefer balance" but "*search* the split, balance-first": seed the branch-and-bound with the balanced allocation (an excellent, usually-optimal incumbent) and let an admissible lower bound prune the rest. When a known upper bound for the target exists — typically the Kronecker rank of a factorisation $\langle n,m,p\rangle=\langle n_1,m_1,p_1\rangle\otimes\langle n_2,m_2,p_2\rangle$ — it is used as the incumbent: if the base's root lower bound already meets or exceeds it, the base cannot improve and the allocation sweep is dropped outright.

9.5 Open questions

Two extrapolations are untested here. (i) *Scale*: the catalog thins above max-dim 16, so Table 3 cannot yet say whether the average ω-vs-imbalance slope persists, flattens, or reverses for large N; the asymptotic τ-theorem constructions suggest the unbalanced tail should only get richer. (ii) *Rectangular targets*: the metric above folds rectangular shapes into a single imbalance scalar

$p - n$; a genuinely rectangular sweep (independent control of $m - n$ and $p - m$) may expose direction-dependent structure the cubic-centred view misses.

10 Comparison with the community catalogs

Our catalog's comparison refreshes that of Drevet–Islam–Schost 2009 ("DIS09" [3]; Tables 3 and 4 in the original numbering) with the following changes in structure relative to that reference.

- Per field, not per merged cluster. Each of $\mathbb{Z}$, $\mathbb{Q}$, $\mathbb{R}$, $\mathbb{C}$, $\mathbb{F}_2$, $\mathbb{F}_3$ gets its own column — the fields are kept *distinct* (there is no "R/Q/Z" merge) — and ternary-integer ($\mathbb{Z}$T) is surfaced as a per-scheme flag. The inclusion $\mathbb{Z} \subset \mathbb{Q} \subset \mathbb{R} \subset \mathbb{C}$ is used only when computing *which* fields a scheme satisfies (field widening), never to collapse columns.
- Commutative axis split out. DIS09 Table 4 (commutative) is preserved as a separate set of columns alongside the non-commutative ones, so the reader cannot accidentally compare Rosowski 21 to Laderman 23 (Section 2).
- Every cell carries an `attribution_for_rank`, which is the earliest source that established the bound. The contemporary importing source (FMM-Lille, Perminov, AlphaTensor bulk imports) is recorded separately and does not appear in the comparison cells.
- Border-rank entries are split out separately; the rank comparison reports tensor rank only.

10.1 SOTA definition

Throughout the catalog, "SOTA" means the lowest known rank for $(\langle n, m, p\rangle, K, \text{commutative})$ where the rank is witnessed by either an explicit catalog entry, a cited bound, or a derived bound. This is a working definition; several corner cases require explicit decisions, currently captured in `paper/sota-conventions.md`:

Mixed-field. When a scheme is valid over $\mathbb{R}$ but its coefficients all lie in $\mathbb{Z}$, we report it as both an R-cell and a Q-cell. The field-widening sweep drives the cross-population.

Border vs exact. Border-rank bounds are excluded from the tensor-rank comparison and appear in the border-rank appendix only.

Partially verified. Cited bounds that lack a verified scheme are flagged with a dagger (†) in the cell.

$\mathbb{F}_2/\mathbb{F}_3$. Integer schemes reduce mod 2 and mod 3; the field stampers grant $\mathbb{F}_2/\mathbb{F}_3$ membership systematically (Section 2).

10.2 Live comparison versus date-stamped snapshot

The per-shape rank landscape is *actively moving*: lower-rank schemes for fixed formats appear on a near-continuous basis — from this project's own search, from the external catalogs we track (FMM-Lille [24], Perminov [16]), and from learning-based efforts (AlphaTensor, AlphaEvolve). Any static per-shape rank table would be stale before the paper appears. The exhaustive, continuously-regenerated comparison therefore lives in the catalog browser at `solven.eu/matmulcatalog` (machine twins under `docs/comparison/`), which is rebuilt from the catalog on every change and is the source of truth for any specific rank.

What the paper prints instead is a *date-stamped snapshot* (2026-07-12 throughout this section), clearly labelled as such and regenerable by a single command — enough to characterise *how* the pipeline compares against the community catalogs without freezing a leaderboard. Every number below is machine-emitted from the same JSON the browser renders.

10.3 Head-to-head over $\mathbb{Q}$

We compare over $\mathbb{Q}$ — the field where both external catalogs carry the most data, and our sweeps' default field. Per shape we track our best *derived* rank (produced by the search of Section 8) separately from our best *imported* rank (an external scheme we carry), so that "we hold the best known scheme" and "our own derivation produced the best known rank" are never conflated. As of 2026-07-12, over 5440 comparable shapes (max dimension ≤ 32):

our derived rank strictly beats every external holding	908
our derived rank ties the external best	3688
we hold the external best only as an import	184
an external catalog strictly beats everything we hold	660

Table 6 lists the largest-margin wins; the 908 wins carry margins totalling 20,801 ranks (median 14, maximum 199).

Table 6: Shapes where this work's own *derived* non-commutative rank over $\mathbb{Q}$ strictly improves on every external holding (FMM-Lille, Perminov). The *carry* column is the best rank we hold as an import over the same field; the *operator* column is the root of our derived scheme's lineage (see Table 7). Top 24 of 908 such shapes by margin; full data in `docs/comparison/us-vs-fmm-vs-perminov-Q.json`.

shape	this work	carry	FMM-Lille	Perminov	operator
$\langle 29, 31, 32\rangle$	**14506**	—	14705	—	projection
$\langle 25, 32, 32\rangle$	**13356**	—	13533	—	recombination
$\langle 30, 31, 32\rangle$	**14682**	—	14841	—	projection
$\langle 25, 27, 28\rangle$	**10006**	—	10164	—	projection
$\langle 28, 29, 30\rangle$	**12285**	—	12433	—	projection
$\langle 21, 25, 31\rangle$	**8899**	—	9040	—	recombination
$\langle 26, 27, 28\rangle$	**10160**	—	10297	—	projection
$\langle 30, 31, 31\rangle$	**14369**	—	14502	—	projection
$\langle 27, 29, 29\rangle$	**11820**	—	11952	—	projection
$\langle 15, 28, 31\rangle$	**7134**	—	7266	—	recombination
$\langle 22, 23, 29\rangle$	**8074**	—	8205	—	recombination
$\langle 24, 25, 26\rangle$	**8291**	—	8417	—	projection
$\langle 28, 29, 29\rangle$	**12005**	—	12131	—	projection
$\langle 22, 25, 31\rangle$	**9302**	—	9424	—	recombination
$\langle 27, 27, 32\rangle$	**12150**	—	12274	12270	recombination
$\langle 20, 27, 31\rangle$	**9080**	—	9198	—	recombination
$\langle 26, 27, 27\rangle$	**9913**	—	10030	—	projection
$\langle 18, 22, 31\rangle$	**6745**	—	6852	—	recombination
$\langle 31, 31, 32\rangle$	**14836**	—	14940	—	projection
$\langle 21, 24, 31\rangle$	**8386**	—	8488	—	recombination
$\langle 13, 15, 30\rangle$	**3390**	—	3490	—	recombination
$\langle 17, 21, 31\rangle$	**6189**	—	6287	—	recombination
$\langle 29, 29, 30\rangle$	**12429**	—	12526	—	projection
$\langle 29, 29, 31\rangle$	**13270**	—	13367	—	concatenation

Perminov ties are not independent confirmations. The comparison must be read with the data-flow direction in mind: Perminov's catalog *ingests* community results, including ours — as of 2026-07-12 it redistributes 14 of our scheme files across 11 shapes (credited under

`schemes/known/matmulcatalog/`), all of them his best holding for the field, all from our Hopcroft–Kerr program (Section 7). A tie with Perminov may therefore be our own scheme round-tripped. FMM-Lille likewise updates its index in response to community results (Section 10.5 and the $\langle 17, 17, 17\rangle$ history in Section 5). This is the attribution discipline of Section 3 applied in reverse: we date-stamp our claims because the catalogs we compare against absorb them.

10.4 Anatomy of the wins

Where do the 908 wins come from? Table 7 classifies each winning scheme by the root operator of its recorded lineage.

Table 7: Anatomy of the 908 strict wins over $\mathbb{Q}$ (Table 6): root operator of the winning scheme's lineage, and the most frequent combining bases among winning recombinations (orientation variants folded). Of the recombination-rooted wins, 762 use *unbalanced* allocations and 7 balanced ones. Margins over the best external holding: 20801 ranks in total, median 14, maximum 199.

root operator	wins	top combining base	wins
recombination with allocation	769	$\langle 2, 2, 2\rangle$	295
axis concatenation	48	$\langle 2, 4, 4\rangle$	150
downward projection	39	$\langle 2, 3, 3\rangle$	70
serendipitous product	29	$\langle 2, 5, 5\rangle$	68
contraction sum	22	$\langle 3, 4, 5\rangle$	22
Kronecker product	1	$\langle 4, 4, 5\rangle$	20

Three findings stand out.

- **Recombination with unbalanced allocations is the engine.** 769/908 wins (85%) are recombination-rooted — and of those, 762 use *unbalanced* block splits against 7 balanced ones. The balance study of Section 9 is thereby confirmed at census scale: the records live where the split is uneven.
- **Small, well-oriented bases dominate.** The winning combining bases are overwhelmingly the small $\langle 2, \cdot, \cdot\rangle$ schemes — $\langle 2, 2, 2\rangle$ (Strassen/Winograd), $\langle 2, 4, 4\rangle$, $\langle 2, 3, 3\rangle$, $\langle 2, 5, 5\rangle$ — frequently under a non-identity axis orientation. Breadth of the base pool and the axis-flip/orientation orbit (Section 4.3.2), not any single clever base, is what pays.
- **The reducing operators supply the long tail.** Downward projection roots 39 wins, the serendipitous product 29, axis concatenation 48, contraction sums 22 — individually modest, collectively the difference between a good catalog and the frontier. All of them also appear as *inner* nodes of recombination-rooted wins.

10.5 Anatomy of the losses

Honesty requires the same census in the other direction, and it decomposes into three very different kinds of "behind".

Against FMM-Lille's raw index. Over the shapes FMM covers, we are better on 1476, tied on 3924, and behind on 26 (total deficit 367 ranks). But the 26 must be audited before being read as a to-do list: we downloaded and rank-counted every artifact behind them. For 20 of the 26, the published artifact does *not* attain the index rank — the file realises a higher rank than the page claims (e.g. $\langle 22, 23, 23\rangle$: index 6435, artifact 6501), is a placeholder containing no tensor data ($\langle 16, 20, 28\rangle$, $\langle 16, 20, 29\rangle$, $\langle 20, 24, 25\rangle$), or carries a construction description whose arithmetic does not check out. These rows are *upstream-unverifiable*: we exclude them from the deficit rather than chase numbers no published scheme witnesses (findings reported to the maintainers).

The audited deficit: six shapes, twelve ranks. What remains after the audit is six shapes where FMM's *verified* artifacts genuinely beat us: four at -1, one at -2, one at -6 — twelve ranks in total, against the 1476 shapes where we lead. Every one of the six traces to the fusion-kernel device of Section 5: cross-slot product sharing that our composition operators deliberately do not perform. The campaign verified exhaustively that no remaining row is closable by existing engines or their parameterisations; closing them requires the new constructor flagged in Section 11. The boundary between composition and hand-crafted sharing is, at this snapshot, worth exactly twelve ranks over thousands of shapes.

Against Perminov. The remaining 634 loss rows are all Perminov-side and of a different nature entirely: they are *published* schemes (flip-graph outputs with explicit factor matrices) that we have not ingested — an import backlog, not a derivation gap. Each is closable by carrying the external scheme; none witnesses a construction technique our operators lack. We keep them in the loss column nonetheless: a comparison that quietly assumed every importable scheme as "ours" would make the win column meaningless.

11 Open questions and future work

We close with what the catalog cannot currently answer and where the next bits of attention should go.

Fusion-kernel extraction. The 2026-07 gap-closing campaign against FMM-Lille (Section 10) ended with a sharp diagnosis: every remaining verified deficit — six shapes, twelve ranks in total — traces to a single device our composition operators do not have, the product-level *fusion kernel* of Section 5: a handful of mixed products serving two or more recursion slots at once. Extracting a reusable identity from the readable specimens (one two-slot unit computes two $\langle 2,3,3\rangle$ slot-problems with 29 products instead of 30), deciding whether the unit is self-contained or relies on whole-kernel cancellation, and turning it into a constructor the allocation search can elect, is the concrete next research program. Notably, disjoint-sum (τ-theorem) search is *not* the missing device: gaps we initially attributed to τ (including $\mathbb{Q}\langle 17,17,17\rangle$, where our recombination search reaches 2930) closed via ordinary recombination, and the campaign verified exhaustively that no remaining row is closable by existing engines or their parameterisations. A constructive disjoint-sum operator retains a narrower motivation: *reproducing* the imported large-format Schwartz–Zwecher schemes rather than carrying them as opaque matrices.

Salt-tolerant prediction. Any post-construction sharing discovery would break the search-rank-equals-materialised-rank invariant (Section 5); if added, it belongs in a separate polish phase after the closure converges, with the interaction with lazy materialisation worked out first.

Border-rank track. The catalog is exact-rank only. A border-rank track would admit results like Bini's $\mathbb{R}\langle 2,2,2\rangle$ border-rank-7 constructions and Schönhage τ-theorem upper bounds; a separate border-rank listing already feeds the catalog browser but not yet the comparison tables.

Lower-bound integration. The lower-bound registry — Strassen's $\mathbb{R}\langle 2,2,2\rangle \geq 7$, Bläser's family, the recent Wang 2026 $\mathbb{F}_2\langle 3,3,3\rangle \geq 20$ — is not yet surfaced in the comparison tables; a gap-to-lower-bound column ("49, gap 4 to lower-bound 45") would make the SOTA frontier read more cleanly.

Border between catalog and hand-crafted-discovery. When a hand-crafted construction lands at a shape where composition was *also* predicting the same rank, the lineage should record

both discoveries and the comparison table should report both; today the catalog picks one. A two-attribution column would force the distinction into the table.

Reproducibility tooling. The paper now prints a date-stamped snapshot of the comparison; the live catalog browser remains the source of truth, regenerated from the catalog and the cited- and derived-bound registries per the conventions of Section 10.1. The remaining work item is finalising the SOTA conventions document, so the comparison has a deterministic rule for every corner case — the prerequisite for the exhaustive SOTA review that the curated paper examples wait on.

12 On the role of the AI coding agent

This catalog, its search engine, and much of this paper were developed in close collaboration with AI agents — primarily Anthropic's Claude via Claude Code, with part of the theoretical investigation carried out with OpenAI's ChatGPT 5.5. We record the division of labour explicitly — in the same spirit as the optimality discipline of the rest of the paper: say exactly what was done, and how confidently it can be attributed.

A fairly clean split emerged between *direction* and *execution*: the human author supplied the questions, the framing, and final judgement; the agent supplied most of the execution — implementation, routine derivations, test scaffolding, provenance bookkeeping — and a substantial share of the *theory*, but only once a direction was fixed. Two concrete cases, with our (necessarily subjective) attribution:

- **The recombination branch-and-bound** — the allocation/orbit search that assigns target dimensions to a base's slots and prunes by a rank bound — was roughly **95% agent**: both the idea to cast allocation as a bounded search and its implementation.
- **The exhaustive enumeration of recombination multisets** (Section 6): the *theory* — the per-axis decoupling lemma that turns an intractable $39k^7$ search into an exact, direction-stratified enumeration — and its implementation were again roughly **95% agent**. But the *idea to investigate the multiset perspective at all* was **100% the human author**: the agent did not propose that this was a fruitful lens, it only developed and executed it once asked.

A second, less flattering pattern was just as consistent: the agent **hallucinated frequently** — confident, plausible, and *wrong* intermediate claims: false "facts", derivations that did not hold, bugs reported as fixes. The most telling instance was recurring: asked to explain how the Université de Lille FMM catalog attains a particular finite-size rank, the agent would repeatedly reach for Schönhage's τ-theorem (the asymptotic sum inequality) [21] as the mechanism — something this project neither runs nor derives, and which does not produce the concrete finite schemes in question (those come from FMM's own recursive concatenation and projection). This confabulation was strikingly persistent: it recurred across sessions and survived explicit standing instructions recorded in the agent's persistent project memory that warn against exactly it, requiring the human to steer the agent off it hard and often. That written, recalled-in-context rules did not reliably suppress a favoured hallucination is itself worth recording. The consequence is structural: *every* agent-produced claim in this work was gated behind an independent check — a verifier, an exhaustive recomputation, or an adversarial second pass — and the project's emphasis on cheap, automatic verification (Section 5) is in part a direct response. Hallucination was the norm, not the exception; what made the collaboration productive was not trusting the agent but *cheaply catching* it.

A cross-model data point. The Hopcroft–Kerr $\langle 2, p, n\rangle$ constructive problem (Section 7) doubles as a controlled comparison across model generations: three distinct agents attacked

the *same* problem with the same repository context. OpenAI's ChatGPT 5.5 (theory-first, Mathematica) and Anthropic's Claude Opus 4.8 (implementation-first) both stalled — between them, partial emitters, a correct-but-narrow impossibility theorem, and no formula-attaining rectangular scheme; for months the HK schemes stood in the catalog as *imported but not re-derived*. Claude Fable 5 then broke the problem in a few hours of session time, overturning the operative impossibility over the true reusable set and emitting the machine-verified schemes — including $\langle 2, 10, 15\rangle$=233, below every published catalog. The human contribution remained the direction ("crack HK71 constructively"), the paper PDF, and the acceptance tests; the chain of mathematical moves was the agent's — a single data point, not a benchmark, but the clearest capability step we have observed between model generations on this project.

Tooling. The bulk of the work was carried out with Anthropic's Claude Code on Claude Opus 4.7 and 4.8; the most recent work (including parts of this section) runs on Claude Fable 5. The hallucination rate noted above is for the Opus 4.x models specifically, and is a moving target.

The percentages are self-assessments, not measurements — an upper bound on the author's share of the keystrokes, not a proven division of credit. The split they sketch is the useful data point: the agent is strongest at turning a well-posed direction into correct theory, code, and catalog-scale breadth; the scarce, human-supplied ingredient is the choice of question. Human chooses the lens, agent builds the optics.

A Worked product tables: Strassen and Strassen–Winograd

The seven products of the Strassen and Winograd rank-7 $\langle 2, 2, 2\rangle$ templates over a heterogeneous split $\langle M_1{+}M_2, N_1{+}N_2, P_1{+}P_2\rangle$, with block "11" the larger part on every axis (real data top-left, any padding at the bottom/right). The four A-blocks have shapes $A_{11}\colon M_1 \times N_1$, $A_{12}\colon M_1 \times N_2$, $A_{21}\colon M_2 \times N_1$, $A_{22}\colon M_2 \times N_2$, and likewise for B and C; each product is a (sum of A-blocks) $\cdot$ (sum of B-blocks), realised at its genuine nonzero overlap under the zero-sparing rule of Section 4.3 ("blk" = single block, "sum" = two-block sum).

Strassen		operands	realised size
M_1	$(A_{11}{+}A_{22})(B_{11}{+}B_{22})$	sum·sum	$\langle M_1, N_1, P_1\rangle$
M_2	$(A_{21}{+}A_{22})\,B_{11}$	sum·blk	$\langle M_2, N_1, P_1\rangle$
M_3	$A_{11}(B_{12}{-}B_{22})$	blk·sum	$\langle M_1, N_1, P_2\rangle$
M_4	$A_{22}(B_{21}{-}B_{11})$	blk·sum	$\langle M_2, N_2, P_1\rangle$
M_5	$(A_{11}{+}A_{12})\,B_{22}$	sum·blk	$\langle M_1, N_2, P_2\rangle$
M_6	$(A_{21}{-}A_{11})(B_{11}{+}B_{12})$	sum·sum	$\langle M_1, N_1, P_1\rangle$
M_7	$(A_{12}{-}A_{22})(B_{21}{+}B_{22})$	sum·sum	$\langle M_1, N_2, P_1\rangle$

Winograd		operands	realised size
P_1	$A_{11}\,B_{11}$	blk·blk	$\langle M_1, N_1, P_1\rangle$
P_2	$A_{12}\,B_{21}$	blk·blk	$\langle M_1, N_2, P_1\rangle$
P_3	$(A_{21}{+}A_{22})(B_{12}{-}B_{11})$	sum·sum	$\langle M_2, N_1, P_1\rangle$
P_4	$S_2\,T_2$	sum·sum	$\langle M_1, N_1, P_1\rangle$
P_5	$S_3\,T_3$	sum·sum	$\langle M_1, N_1, P_2\rangle$
P_6	$S_4\,B_{22}$	sum·blk	$\langle M_1, N_2, P_2\rangle$
P_7	$A_{22}\,T_4$	blk·sum	$\langle M_2, N_2, P_1\rangle$

with $S_2{=}A_{21}{+}A_{22}{-}A_{11}$, $S_3{=}A_{11}{-}A_{21}$, $S_4{=}A_{11}{+}A_{12}{-}A_{21}{-}A_{22}$, $T_2{=}B_{11}{-}B_{12}{+}B_{22}$, $T_3{=}B_{22}{-}B_{12}$, $T_4{=}B_{11}{-}B_{12}{-}B_{21}{+}B_{22}$.

B Implementation notes

Concept-to-code map and engineering notes for the search engine of Section 8.

Concept-to-class map. The main concepts of Section 8 correspond to the following classes and flags in the reference implementation:

concept	class / flag
frontier driver (impact-DAG priority queue)	`FrontierClosure`
recombination materialiser	`Recombination.constructWithAllocation`
composed sub-product lookup (overlay + disk)	`AlgorithmLookup`, `FieldAwareLookup`
allocation branch-and-bound	`AllocationOptimizer`
search budget	`SearchBudget`: `EXACT`, `upTo`, `maxNodes`
allocation filters	`maxImbalance`, `maxCombinations`
axis-flip orbit mask evaluation	`AnalyticalMaskSearch`
seed ranking by ω	`docs/bases-by-omega.md`

Pool configuration. The default pool (`PoolConfig.simple`) carries a small number of high-value outer schemes — Strassen, Winograd, Smirnov 3-row templates, and a few imported Pan / AT / FMM bases. An `extendedPool` mode adds every `Leaf`-tagged catalog entry as an outer template; that pool is large enough that the `maxCombinations`-capped allocation enumerator becomes load-bearing.

Budget flags. `AllocationOptimizer` runs under a `SearchBudget`: `EXACT` (no cap), `upTo` a known upper bound (e.g. the Kronecker rank), or a `maxNodes` anytime cap. The result carries an `exhaustive` flag — propagated to the optimality tier of Section 1 — and reports the `nodes` visited against the `fullSpace` of allocations.

Seed ranking. `docs/bases-by-omega.md` is an automatically-regenerated table ranking every pool-eligible base by its raw ω, its ω after imbalance-≤ 3 restriction, and its ω after axis-flip-canonical and S_3-canonical orbit reductions, evaluated at the reference target $\langle 17, 17, 17\rangle$.

Caching. Caching exists at two layers. The first is per-scheme *symbolic templates*: for each outer scheme we extract once a tuple of six block-support bitmasks per product (U-row, U-col, V-row, V-col, W-row, W-col), and per-call sub-shape extraction reduces from $O(r \cdot \dim^2)$ to $O(r \cdot \dim)$. The second is a per-(scheme, allocation, peel) cache of the resulting sub-shape multiset; this is a hashing-overhead trade and is currently capped at 200k entries per scheme. A symbolic-template-at-allocation-pattern layer would generalise the second cache; the current implementation captures the common cases without paying the templating cost.

Progress. A derivation closure over the entire catalog up to dim 32 takes hours of wall-clock; long runs emit periodic progress lines and ETAs at three nested levels (per-round, per-shape, and intra-search).

References

[1] Vladimir P. Burichenko. On symmetries of the Strassen algorithm. *arXiv:1408.6273*, 2014.

[2] Hans F. de Groote. On varieties of optimal algorithms for the computation of bilinear mappings. ii. optimal algorithms for 2×2-matrix multiplication. *Theoretical Computer Science*, 7(2):127–148, 1978.

[3] Charles-Éric Drevet, Md. Nazrul Islam, and Éric Schost. Optimization techniques for small matrix multiplication. *Theoretical Computer Science*, 412(22):2219–2236, 2011. Preprint dated 2009; published 2011.

[4] Jean-Guillaume Dumas, Clément Pernet, and Alexandre Sedoglavic. A non-commutative algorithm for multiplying 4×4 matrices using 48 non-complex multiplications, 2025. arXiv:2506.13242.

[5] Alhussein Fawzi et al. Discovering faster matrix multiplication algorithms with reinforcement learning. *Nature*, 610:47–53, 2022.

[6] John E. Hopcroft and Leslie R. Kerr. On minimizing the number of multiplications necessary for matrix multiplication. *SIAM Journal on Applied Mathematics*, 20(1):30–36, 1971.

[7] Md. Anwarul Islam. Bilinear algorithms for matrix multiplication and their applications. *PhD thesis, University of Western Ontario*, 2009.

[8] I. E. Kaporin. Semi-analytical solution of brent equations. *Doklady Mathematics*, 518(1):29–34, 2024.

[9] Manuel Kauers, Jakob Moosbauer, and Isaac Wood. Exploiting the structure in tensor decompositions for matrix multiplication. *arXiv preprint arXiv:2602.11041*, 2026.

[10] Kirill Khoruzhii, Patrick Gelß, and Sebastian Pokutta. Local improvements to trilinear aggregation. `https://github.com/khoruzhii/lita`, 2026.

[11] Julian D. Laderman. A noncommutative algorithm for multiplying 3×3 matrices using 23 multiplications. *Bulletin of the AMS*, 82(1):126–128, 1976.

[12] O. M. Makarov. An algorithm for multiplication of 3×3 matrices. *Zh. Vychisl. Mat. i Mat. Fiz.*, 26(2):293–294, 320, 1986.

[13] Alexander Novikov et al. Alphaevolve: a coding agent for scientific and algorithmic discovery, 2025. Google DeepMind. arXiv:2506.13131.

[14] V. Ya. Pan. Strassen's algorithm is not optimal – trilinear technique of aggregating, uniting and canceling for constructing fast algorithms for matrix operations. In *Proc. 19th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 166–176, 1978.

[15] V. Ya. Pan. New fast algorithms for matrix operations. *SIAM Journal on Computing*, 9(2):321–342, 1980.

[16] Andrew I. Perminov. Fastmatrixmultiplication. `https://github.com/dronperminov/FastMatrixMultiplication`. Accessed 2026-06.

[17] Andrew I. Perminov. Fast matrix multiplication via ternary meta flip graphs. *arXiv preprint arXiv:2511.20317*, 2025.

[18] Andrew I. Perminov. Fast matrix multiplication in small formats: Discovering new schemes with an open-source flip graph framework. *arXiv preprint arXiv:2603.02398*, 2026.

[19] Andrew I. Perminov. Meta flip graph meets serendipitous product: new fast matrix multiplication results. *arXiv preprint arXiv:2606.02480*, 2026.

[20] Andreas Rosowski. Fast commutative matrix algorithms. *arXiv preprint arXiv:1904.07683*, 2019.

[21] Arnold Schönhage. Partial and total matrix multiplication. *SIAM Journal on Computing*, 10(3):434–455, 1981.

[22] Oded Schwartz and Eyal Zwecher. Disjoint-sum constructions for fast matrix multiplication. *hal-05121550, preprint*, 2025.

[23] Alexandre Sedoglavic. A non-commutative algorithm for multiplying 7×7 matrices using 250 multiplications, 2017. hal-01572046v2.

[24] Alexandre Sedoglavic et al. Fmm: Fast matrix multiplication catalog. `https://fmm.univ-lille.fr/`. Accessed 2026-06.

[25] Alexey Vladimirovich Smirnov. *Bilinear Complexity and Practical Algorithms for Matrix Multiplication*. PhD thesis, Computing Center of the Russian Academy of Sciences, 1986.

[26] Alexey Vladimirovich Smirnov. Bilinear complexity and practical algorithms for matrix multiplication. *Computational Mathematics and Mathematical Physics*, 53(12):1781–1795, 2013.

[27] Warren D. Smith. Fast matrix multiplication formulae — report of the prospectors. Manuscript, 2002.

[28] Volker Strassen. Gaussian elimination is not optimal. *Numerische Mathematik*, 13(4):354–356, 1969.

[29] Abraham Waksman. On winograd's algorithm for inner products. *IEEE Transactions on Computers*, C-19(4):360–361, 1970.