The covering number $C(13, 7, 4)$ is 30

A classification-assisted exact proof

John Tripodo
ORCID: 0009-0006-9180-4953

29 October 2026

hexagon:2610.00669v1

Abstract

We prove that $C(13, 7, 4) = 30$, using the classification of minimum $(12, 6, 3)$ coverings by Gordon, Patashnik, Petro and Taylor. A hypothetical 29-block cover has at least five points lying in exactly 15 blocks. The link at one of these points leaves only 14 blocks outside it. The other four points force complete families of outside blocks whose overlaps must agree. An exhaustive computation shows that every compatible choice requires 15 outside blocks, giving a contradiction. The known 30-block construction is therefore optimal. We give reproducible exact computations and a public Lean formalization of a second argument for the finite obstruction. The Lean theorem states the published minimum bound and classification as explicit hypotheses; it does not reprove those results.

Keywords: covering designs; covering numbers; finite classification; exact enumeration; computational proof; Lean formalization.

1 The result and its context

For positive integers $v \geq k \geq t$, a (v, k, t) *covering* is a family $\mathcal{F} \subseteq \binom{X}{k}$, where $|X| = v$, such that every $T \in \binom{X}{t}$ is contained in some $B \in \mathcal{F}$. Its minimum cardinality is the covering number $C(v, k, t)$. The problem considered here asks for the minimum number of seven-element blocks on thirteen points needed to cover every four-element subset.

Theorem 1.1 (Main result)**.**

$$\boxed{C(13, 7, 4) = 30.}$$

An explicit construction supplies the upper bound 30; the contribution is the global exclusion of 29 blocks. Prior tabulations gave the interval $28 \leq C(13, 7, 4) \leq 30$ [2], and a separately published, formally checked elimination of one minimum-link type does not resolve all 68 types [3]. Krug recently determined the different covering number $C(12, 6, 4) = 41$ [4].

The one external mathematical ingredient is the work of Gordon, Patashnik, Petro and Taylor [1]: their Theorem 4.4 proves $C(12, 6, 3) = 15$, and their Theorem 5.9 classifies the minimum $(12, 6, 3)$ covers by a Figure 1 completion family and a single exceptional Figure 6 cover. Their 68 nonisomorphic minimum covers are *input*, not an original classification claimed here. The proof below explains why that classification is enough and isolates the precise role of the computation.

Outline of the proof. Every point of a hypothetical 29-block covering occurs in at least 15 blocks, and there are only eight degree units above that baseline; hence at least five points have degree exactly 15. Root at one such point. Its 15 incident blocks determine a classified link, leaving just 14 blocks not containing the root. Each of four further minimum points forces a full set of outside blocks. The classification supplies exhaustive finite domains for those sets. The computation proves that four mutually compatible choices always need at least 15 outside blocks. This is the contradiction.

2 Why five minimum points are unavoidable

Write $X = Y \cup \{\infty\}$ with $|Y| = 12$ once a root is chosen. For any point $x \in X$, define its degree and link by

$$d_{\mathcal{F}}(x) := |\{B \in \mathcal{F} : x \in B\}|, \qquad L_x(\mathcal{F}) := \{B \setminus \{x\} : B \in \mathcal{F},\ x \in B\}.$$

If $\mathcal{F}$ covers every four-set of X, then $L_x(\mathcal{F})$ covers every three-set of $X \setminus \{x\}$. The cited minimum-cover theorem therefore implies $d_{\mathcal{F}}(x) \geq 15$ for every x.

Lemma 2.1 (Five minimum points)**.** *If a* $(13, 7, 4)$ *covering has exactly* 29 *distinct blocks, at least five points have degree* 15*.*

Proof. Double-count incidences between points and blocks:

$$\sum_{x \in X} d_{\mathcal{F}}(x) = 7|\mathcal{F}| = 7 \cdot 29 = 203 = 13 \cdot 15 + 8.$$

Each degree exceeds 15 by a nonnegative integer. At most eight points can have positive excess, leaving at least five of degree 15. □

Choose such a point ∞. The root link $L := L_\infty(\mathcal{F})$ has exactly 15 blocks and is one of the published minimum $(12, 6, 3)$ covers up to relabelling. The *outside family*

$$\mathcal{F}^\circ := \{B \in \mathcal{F} : \infty \notin B\}$$

contains exactly $29 - 15 = 14$ blocks, all seven-subsets of Y. Choose four other degree-15 points $Q \subseteq Y$, $|Q| = 4$. We now show how they constrain $\mathcal{F}^\circ$.

Why use several minimum points? A single minimum point gives a 15-block link but leaves a difficult extension problem on the other twelve points. Every additional minimum point specifies *all* outside blocks through that point. Consequently these local specifications must agree on every shared block. Their union gives a lower bound on the number of outside blocks. The useful obstruction here is the gap $15 > 14$ obtained from four such specifications. It is an obstruction to compatible local links, not a new construction of the upper-bound cover.

3 From classified links to complete outside pencils

Let $\mathcal{C}$ be a set of representatives of the 68 published isomorphism classes of minimum $(12, 6, 3)$ coverings, each with 15 distinct six-subsets of a labelled twelve-element set. Fix a labelled root link $L \in \mathcal{C}$. For $q \in Y$, the root-containing part of the q-link is already fixed:

$$S_L(q) := \{\{\infty\} \cup (A \setminus \{q\}) : A \in L,\ q \in A\}. \tag{1}$$

This is a family of six-subsets of $X \setminus \{q\}$.

Take a classified source link $K \in \mathcal{C}$ on a twelve-point set W, a source point $c \in W$, and a bijection $\phi : W \to X \setminus \{q\}$ such that $\phi(c) = \infty$ and

$$\{\phi(A) : A \in K,\ c \in A\} = S_L(q). \tag{2}$$

Define the associated *complete outside pencil*

$$P(L, q; K, c, \phi) := \{\{q\} \cup \phi(A) : A \in K,\ c \notin A\} \subseteq \binom{Y}{7}. \tag{3}$$

The domain $D_L(q)$ consists of all pencils in (3) over all K, c, ϕ satisfying (2). In computation we may first prune c by the necessary equality $d_K(c) = d_L(q)$, but the definition of the domain itself is unrestricted.

Lemma 3.1 (Domain completeness)**.** *Suppose $\mathcal{F}$ is a 29-block $(13, 7, 4)$ covering with root link L and $d_{\mathcal{F}}(q) = 15$. Then*

$$P_q := \{B \in \mathcal{F}^\circ : q \in B\} \in D_L(q).$$

In particular, P_q records every *outside block through q, not an arbitrarily chosen subfamily.*

Proof. The full link $L_q(\mathcal{F})$ is a minimum $(12, 6, 3)$ cover. The cited classification gives a source $K \in \mathcal{C}$ and a bijection from its twelve points onto $X \setminus \{q\}$ carrying K to $L_q(\mathcal{F})$. If c maps to ∞, the blocks of K containing c map precisely to the known star $S_L(q)$. The remaining blocks of K are exactly the blocks of the q-link that avoid ∞. Reattaching q to their images gives (3), which is exactly P_q. □

The enumeration searches all bijections on the remaining eleven source points that satisfy the star equality. This is needed for Lemma 3.1: searching only a chosen automorphism subgroup would not suffice.

Equivariance of domain reuse. The implementations may reuse a domain when two known stars are isomorphic. This does not require choosing every isomorphism between the target stars. If g is one such isomorphism, extend it by mapping the distinguished target point to the new distinguished target point. Composition with g bijects every classified source embedding for the first star with an embedding for the second, with inverse composition by g^{-1}. Applying this bijection to all complete pencils proves that the transported domain is the entire second domain. Equality of coarse invariants alone is never used to authorize transport.

4 Agreement conditions and the finite obstruction

For distinct $q, r \in Q$ and pencils $P \in D_L(q)$, $R \in D_L(r)$, call (P, R) *compatible* when

$$\{B \in P : r \in B\} = \{B \in R : q \in B\}. \tag{4}$$

The two sides are the outside seven-blocks through both points. Thus the actual pencils P_q, P_r supplied by Lemma 3.1 are pairwise compatible. Also, because each $P_q \subseteq \mathcal{F}^\circ$,

$$\left| \bigcup_{q \in Q} P_q \right| \leq |\mathcal{F}^\circ| = 14. \tag{5}$$

The central finite statement is the following.

Proposition 4.1 (Exhaustive four-pencil obstruction)**.** *For every $L \in \mathcal{C}$ and every $Q \in \binom{Y}{4}$, there are no choices $P_q \in D_L(q)$, $q \in Q$, which satisfy every pair compatibility equation* (4) *and the union budget* (5)*.*

Exact computer-assisted verification. The exhaustive input space contains $68\binom{12}{4} = 68 \cdot 495 = 33{,}660$ *labelled* pairs (L, Q). No orbit reduction of the Q-sets is needed for this claim.

The computation first rejects an input if some pair of selected points has no compatible pair of possible pencils. It then enforces arc consistency: any pencil with no compatible partner in another selected domain is removed, and this rule repeats until stable. These are sound necessary filters. To see the latter formally, consider a putative compatible four-pencil choice. Initially each actual choice lies in its full domain. If all four remain after a round, each is a compatible partner for each of the other three, so none can be removed in the next round. By induction a genuine compatible choice is never eliminated.

The recorded exact census is:

Exhaustive stage	Labelled root/fourset inputs
A pair has no compatible pencil choices	31,263
Additional inputs closed by sound arc consistency	2,355
Inputs retained for the final four-way join	42
Total: $68\binom{12}{4}$	33,660

Table 1: Partition of all labelled root/four-set inputs in the finite search.

For each of the 42 residual inputs, a separately written routine constructs all six pair-compatibility relations and enumerates every four-pencil tuple satisfying them. This final routine *does not prune by union size*; it computes the literal union after joining. Its result is:

Root type	Residual inputs	Compatible tuples per input	Total tuples
Figure 1 completion types	37	216	7,992
Exceptional Figure 6 type	5	54,432	272,160
Total	42		280,152

Table 2: Independent unpruned join. Every one of the 280,152 compatible tuples has a union of exactly 15 distinct outside blocks.

The reconstruction of domains uses the 68-link catalog, with an independent route rebuilding all $28^3 = 21{,}952$ literal Figure-1 completions from the published core and short blocks, recovering 67 completion classes under the construction action, and separately including the exceptional Figure-6 link. Both implementations reconstruct $68 \cdot 12 = 816$ root/point domains, agree on the canonical digest of every domain, and obtain Tables 1 and 2. Python ordinary and optimized executions agree; a distinct C++ implementation reproduced the decisive totals. The source, manifests, independent joins and build receipts are itemized in Appendix B. No floating-point optimization status, solver timeout or unverified infeasibility declaration enters the conclusion. Every residual compatible union has $15 > 14$ blocks, so (5) is impossible. □

5 Proof of optimality

Proof of Theorem 1.1. Suppose a covering has at most 29 blocks. Removing duplicate blocks, if any, preserves coverage; we can pad with previously unused seven-subsets to obtain a covering with exactly 29 distinct blocks, since $\binom{13}{7} = 1716$.

By Lemma 2.1, choose five degree-15 points. Relabel one as ∞ and normalize its minimum link to a representative $L \in \mathcal{C}$; this induces a relabelling of the twelve ordinary points. Choose the other four minimum points as Q. Lemma 3.1 supplies one complete outside pencil $P_q \in D_L(q)$ for each $q \in Q$. All six compatibility equations hold, and the pencils jointly occupy at most the 14 outside blocks. This contradicts Proposition 4.1. Therefore $C(13, 7, 4) \geq 30$. The explicit 30-block covering in Appendix A, checked against every one of the $\binom{13}{4} = 715$ four-subsets, supplies $C(13, 7, 4) \leq 30$. □

6 A public Lean proof relative to the published classification

The repository accompanying this paper contains the complete local source closure for a second proof of the lower bound. This proof works directly with actual covering families. It does not use the Python verifier as an oracle and does not merely import a Boolean claim that the Python search succeeded.

6.1 The mathematical interface

The ground type `C13.Point` is `Fin 13`. A family is a finite set of finite subsets. The predicate `IsCoverOn U F k t` requires every block of F to be a k-subset of U and every t-subset of U to lie in a block. Thus the formal conclusion concerns the same simple covering number as Theorem 1.1; duplicate blocks are not counted.

The principal endpoint is

`C13.Global.exact30_after_published_results`.

It proves the following implication, expressed here without Lean syntax:

$$\mathsf{PublishedMinimumBound} \ \wedge \ \mathsf{PublishedClassification1995}$$
$$\Longrightarrow \quad (\exists F \text{ a } (13,7,4) \text{ cover}, \ |F| = 30) \ \wedge \ (\forall F \text{ a } (13,7,4) \text{ cover}, \ |F| \geq 30)\,.$$

The first hypothesis is the lower-bound part of the published Theorem 4.4. The second is the relabelling coverage supplied by Theorem 5.9 together with the minimum value from Theorem 4.4. They are ordinary theorem hypotheses, not additional Lean axioms. They are nevertheless substantive mathematical assumptions. An axiom audit cannot discharge an explicit hypothesis.

The classification predicate is stated using the literal Figure 1 completion family and Figure 6. The source-to-catalog theorem `catalog_covers_all_raw_sources` then covers every one of the $28^3 = 21{,}952$ completion triples and the exceptional source. The needed direction is coverage by the catalog. Pairwise nonisomorphism of catalog entries under every permutation is not needed for the lower bound.

6.2 The actual-cover certificate argument

For a catalog link K and a four-set Q, the predicate `ActualObstruction K Q` excludes an actual global cover whose root link is K, whose points in Q have degree 15, and whose outside family

has at most 14 blocks. The definition retains the classification hypothesis. It is not the stronger statement that every abstract assignment of local pencils is impossible without the global covering constraints.

The formal census uses the same 33,660 labelled positions, with the following distinct partition:

Formal exclusion mechanism	Positions
Empty-support certificates	31,762
Exact integer-dual certificates and checked transports	1,893
Exceptional Figure-6 predecessor certificates	5
Total	33,660

This is not the Python partition of Table 1. The integer-dual branch contains 964 representative positions and 929 positions reached by checked root automorphisms. The empty-support branch uses 3,636 certificates, some of which exclude more than one four-set.

The formal argument includes the full $792 = \binom{12}{7}$ outside block universe, complete-coverer masks, and all $816 = 68 \cdot 12$ root/point alphabet bindings. A block containing none of the selected points is handled separately; it cannot be silently discarded from the outside-block budget. Row-validity theorems derive the covering and incidence constraints from genuine covers before a dual inequality is used. The checked census and `all_catalog_foursets` feed into `no_cover29`; the final theorem combines this with padding and the unconditional 30-block witness.

6.3 Paper-to-Lean correspondence

The following modules are under `lean/` in the public repository. The correspondence is semantic; it does not assert that the Lean certificate search follows the Python search instruction for instruction.

Mathematical step	Formal source
Links, degrees, incidence sum, five minimum points, padding	`C13Covering/Reduction.lean`
Literal historical classification predicate	`C13Covering/SourceFigures.lean`
Coverage of all raw classified sources by the catalog	`C13Covering/OrbitAllSources.lean`
Complete semantic alphabets and unrestricted star transport	`C13RawStarRecovery/StarTranslation.lean`
Actual-cover obstruction, covering of all four-sets, transport	`C13Global/CensusCore.lean`
Validity of exact covering/degree rows and dual implications	`C13Global/RowValidity.lean`, `C13Global/DualBridge.lean`
All 68 roots; no 29-cover; exact-covering endpoint	`C13Global/Final.lean`
Unconditional 30-block witness and model transport	`C13/Known30.lean`, `C13Covering/PositiveBridge.lean`

An additional endpoint, `exact30_after_published_classification`, derives the minimum-bound parameter from the stated all-size-15 classification predicate and checked private-triple irredundancy of every catalog block. This is a valid implication within that formal interface. It is not a claim that the historical proof of Theorem 5.9 is independent of Theorem 4.4, nor a new proof of the 1995 classification.

7 Reproducibility, assurance and limitations

The paper, Lean sources and verification programs are available at

`https://github.com/jjtripodo-lgtm/c13-covering-number`.

The formalization cited here is frozen at public commit

`254ee82fdbc215ad73f732f807a83d69be7c1e22`

The main theorem is directly accessible. The repository also provides a paper-to-proof map, pinned dependencies, and commands for reproducing the checks (Appendix B). No private repository is required.

The Python computations were rerun for this edition in both ordinary and optimized modes, including the independent domain reconstruction and unpruned join. They reproduce Tables 1 and 2. The native C++ implementation agrees. The known upper-bound construction passes all 715 coverage tests; malformed inputs and small brute-force comparisons test the verification routines.

For Lean, the release includes 1,751 unchanged local source modules, authenticated against their original Git objects and recorded source hashes. The preserved historical audit checked 76,982 project declarations in 1,750 modules, allowing only `propext`, `Classical.choice` and `Quot.sound`, with no unsafe project declarations. A later recorded check recompiled the endpoint and ran negative controls using authenticated cached dependencies. These records are not a fresh build of the new public package: that build has not been run in this release environment. The distinction is recorded alongside the source and reproduction instructions.

Both arguments rely on the published minimum bound and classification. The literal input families are printed in Appendix C; the repository preserves the source-transcription audit. The theorem wording was checked against the author-posted preprint for this edition, but a fresh visual comparison of its figures was not completed. This result determines one covering number and does not classify all optimal 30-block covers.

Acknowledgments and use of AI

The author thanks the authors of the published classification and the maintainers of the La Jolla Covering Repository for making their work available. Anthropic Claude and OpenAI ChatGPT/Codex were used extensively in developing the arguments, programs, formalization, and manuscript. The text is primarily AI-generated. John Tripodo directed the project and is responsible for the submitted work. No part of the contribution is attested as having been checked and understood by a human, and no independent human referee review is claimed. The computational checks and recorded Lean audits described above are separate forms of evidence.

A An explicit optimal 30-block covering

The blocks below are the published Nurmela–Östergård upper-bound construction tabulated by the La Jolla Covering Repository [2], not a new construction claimed here. They are seven-subsets of $\{1, 2, \ldots, 13\}$. Each four-subset is contained in at least one of the 30 blocks. The accompanying independent standard-library checker enumerates all 715 possible four-subsets, verifies distinctness and confirms $1{,}050 = 30\binom{7}{4}$ total block/fourset incidences.

Block	Block
$\{1,2,5,6,7,8,9\}$	$\{1,3,4,5,6,8,10\}$
$\{2,3,6,7,8,9,10\}$	$\{2,3,4,5,8,9,11\}$
$\{1,3,5,7,8,10,11\}$	$\{1,2,4,8,9,10,11\}$
$\{1,2,3,5,7,10,12\}$	$\{1,3,4,5,9,10,12\}$
$\{1,2,4,5,6,11,12\}$	$\{1,3,6,7,9,11,12\}$
$\{2,6,7,8,9,11,12\}$	$\{4,6,7,8,9,11,12\}$
$\{2,3,4,6,10,11,12\}$	$\{5,6,7,9,10,11,12\}$
$\{3,4,5,6,7,9,13\}$	$\{1,2,3,5,6,10,13\}$
$\{1,4,6,7,9,10,13\}$	$\{1,2,3,4,7,11,13\}$
$\{2,4,7,8,9,11,13\}$	$\{2,4,5,7,10,11,13\}$
$\{1,3,5,9,10,11,13\}$	$\{1,4,5,7,8,12,13\}$
$\{2,4,6,7,8,12,13\}$	$\{1,2,3,8,9,12,13\}$
$\{3,4,7,8,10,12,13\}$	$\{2,5,8,9,10,12,13\}$
$\{3,5,6,8,11,12,13\}$	$\{2,4,6,9,11,12,13\}$
$\{6,7,8,9,11,12,13\}$	$\{1,6,8,10,11,12,13\}$

The example is different from the separate 30-block cover in `known30_control.json`. Both were checked independently on all 715 four-subsets in the preparation packet.

B Reproducing the computations and formal proof

Download the public repository linked in Section 7. From its top-level directory, run:

```
python3 verify_release.py
python3 computation/reproduce.py
python3 computation/reproduce.py --native
```

The first command checks source hashes, local Lean imports, dependency pins and the preserved historical axiom log; it does not run Lean. The second runs the exact Python proof, checks both ordinary and optimized executions against frozen outputs, and runs the controls. The optional third command adds the C++ and undefined-behaviour-sanitizer cross-checks; it requires a C++17 compiler and OpenSSL development files. The Python checks use only the standard library and run without network access. Each run writes its outputs and a machine-readable receipt to `computation/replay-output/`.

The lower-bound programs are `verify_pencils.py` and `independent_raw_replay.py`; the independent unpruned join is in `peer_union_join.py`. They reside in the repository's `computation/source/research/horizonmath/c13_classified_pencils/` directory. The input catalog, domain digests and expected outputs are listed in the manifests alongside them. These identities distinguish a reproduction of the released computation from a run on altered inputs.

For the formal proof, install the version in `lean/lean-toolchain` and obtain the revision-pinned dependencies in `lean/lake-manifest.json`. Then, from `lean/`, run:

```
lake build C13Global.Final C13Global.Statistics
lake env lean --trust=0 -j 1 -M3072 C13Global/AuditFinal.lean
```

Start with no local build objects to recompile the included proof modules. Using upstream Mathlib caches does not rebuild Mathlib itself; a complete source rebuild must also rebuild the upstream dependencies. The original checks used a 3 GiB per-process memory cap. Resource requirements and the historical check boundaries are documented in `lean/README.md`. The Lean package is not required to reproduce the Python proof.

C Literal classification input

For transcription checking, the Figure 1 core is the following list on $\{0, 1, \ldots, 9, A, B\}$, with $A = 10$ and $B = 11$:

```
014589  0167AB  2345AB  236789
02468A  02579B  13469B  13578A
03478B  03569A  12479A  12568B.
```

To this core add one six-set extending each of 0123, 4567 and $89AB$, choosing the other two points from the complementary eight. The program enumerates all 28^3 choices, including choices equivalent under relabelling. The construction group has order 576 and gives 67 representative completions. For completeness of the search it is sufficient that every literal completion is in one of these orbits; no claim that arbitrary cover isomorphisms preserve the core is needed.

For Figure 6 the labels are $0123456789TE$, with $T = 10$ and $E = 11$:

```
012345  012678  0129TE  0378TE  0456TE
045789  034569  13689E  14579E  14568T
13457T  23679T  24589T  24567E  23458E.
```

These are transcriptions of the cited classification inputs, not a new classification proof. Source-figure comparison and the applicability of the published classification must be checked independently of the code.

D Finite-verifier logic in pseudocode

The following specifies the logical search performed by the exact finite verifier, independently of implementation-level optimizations. Full reproducibility requires the released source and input catalog.

```
for each root link L in the 68-link catalog:
  for each Q among the 495 ordinary foursets:
    build each complete domain D[L,q], q in Q
    if some pair of domains has no compatible pair:  continue
    repeat:  discard values lacking support in another domain
    until no change
    if some domain is empty:  continue
    enumerate all four-tuples matching all six pair relations
    for each compatible tuple, measure its literal block union
    assert union cardinality exceeds 14
```

Arc-consistency deletion is valid by the induction in the proof of Proposition 4.1. Domain construction is exhaustive by Lemma 3.1, *provided* the cited published classification is correct and fully transcribed. The final join enumerates compatible tuples directly; it uses no external infeasibility oracle.

References

[1] D. M. Gordon, O. Patashnik, J. Petro and H. Taylor, *Minimum* $(12, 6, 3)$ *Covers*, Ars Combinatoria **40** (1995), 161–177. `https://combinatorialpress.com/ars-articles/volume-040-ars-articles/minimum-12-6-3-covers/`.

[2] D. M. Gordon, *La Jolla Covering Repository*, $(v, k, t) = (13, 7, 4)$, historic upper and lower bounds. `https://ljcr.dmgordon.org/cover/table.html` and `https://ljcr.dmgordon.org/cover/low_tab.html`.

[3] E. Dwares, *Formal elimination of link class 52 in the* $C(13, 7, 4)$ *search*, public certification project (2026), `https://github.com/edwares/class52-formal-certification`. The certification concerns one class, not the full optimum.

[4] C. Krug, *The covering number* $C(12, 6, 4)$ *is 41*, preprint (2026), arXiv:2607.23766, `https://arxiv.org/abs/2607.23766`.