COMPLETE UNIFORM HYPERGRAPHS THAT SUPPORT AN UNSATISFIABLE CNF: TWO OPEN CELLS, ONE CORRECTION, AND THE THRESHOLD SEQUENCE

KOREA SUPERINTELLIGENCE LABS

Abstract. Karve and Hirani attach to a hypergraph g the set of CNFs that *live* on it — one clause per edge, on exactly that edge's variables, with an arbitrary choice of signs — and call g *totally satisfiable* when every such CNF is satisfiable. For the complete a-uniform hypergraph $K(a,b)$ on b vertices they publish a table of statuses for $a, b \leq 7$, in which two cells are printed `unknown`. We settle both: $K(4,7)$ is not totally satisfiable, by an explicit 35-clause CNF that all 128 assignments falsify, and $K(5,7)$ is totally satisfiable, because $\binom{7}{5}2^2 = 84 < 128$. We also correct a third cell: the table prints $K(3,5)$ as **`unsat`** in boldface, and $K(3,5)$ is in fact totally satisfiable — an exhaustive search over all 8^{10} sign choices for the ten triples of a five-element set finds no unsatisfiable one. Beyond the published range we compute the threshold $t(a) = \min\{b : K(a,b) \text{ is not totally satisfiable}\}$ for $a = 2, \ldots, 8$, obtaining $t = 4, 6, 7, 8, 9, 11, 12$, and we compare it with the elementary counting bound $m(a) = \min\{b : \binom{b}{a} \geq 2^a\}$: the two agree at every computed a except $a = 3$, and $a = 3$ is exactly the cell the published table gets wrong. Every result stated in this abstract is machine-checked in Lean 4, in a development that declares no axiom of its own and contains no `sorry`; the reproduction of the published table and both formerly unknown cells are checked by the kernel outright, the corrected cell and the thresholds for $a \geq 6$ by the compiled evaluator.

1. Introduction

CNFs living on a hypergraph. Karve and Hirani [1] study satisfiability one hypergraph at a time rather than one formula at a time. Their construction sends a (looped, multi-) hypergraph g to a *set* of CNFs: an edge of size k and multiplicity n carries n distinct clauses on exactly that edge's k variables, each clause being one of the 2^k sign patterns available on them, and a CNF *lives on* g when it is assembled from one such choice at every edge ([1, §2.5.1], the maps $f_1, \ldots, f_4$). The definition that organises the subject is then [1, §2.5.2]:

> "A set of Cnfs g is *totally satisfiable* if it is nonempty and if every Cnf in it is satisfiable. Otherwise, it is *unsatisfiable*."

and with it the decision problem the paper is named for [1, §2.5.3]:

> "**Instance**: Given a specific looped-multi-hypergraph g. **Question**: Is every Cnf x such that $x \in g$ satisfiable."

Total satisfiability is a property of the *shape* of a formula — of which variables occur together in a clause — and not of its signs, and it is therefore a statement about an entire class of Boolean formulae at once. The predecessor paper [2] settles the graph case completely: a simple graph, read as a 2-uniform hypergraph, is unsatisfiable if and only if it contains one of four graphs (butterfly, bowtie, K_4, book) as a topological minor. For hypergraphs no such classification is known.

Complete uniform hypergraphs, and the table. Let $K(a,b)$ denote the complete a-uniform hypergraph on b vertices: all $\binom{b}{a}$ subsets of size exactly a, each an edge of multiplicity one ([1, §8.1], where it is written $\mathcal{K}_{a,b}$). A CNF living on $K(a,b)$ is thus a choice of one sign pattern per a-subset, and $K(a,b)$ is totally satisfiable exactly when no such choice is unsatisfiable. Karve and Hirani single these hypergraphs out because they combine two opposing tendencies: "having all possible

TABLE 1. The table of [1, Table 3], transcribed from the `v1` e-print. Boldface is the authors' own mark for a non-obvious result. Entries below the diagonal are printed as dashes there and left blank here, and the source's table carries a further $\cdots$ column and $\vdots$ row marking that it continues.

	$b=1$	$b=2$	$b=3$	$b=4$	$b=5$	$b=6$	$b=7$
$a=1$	sat	sat	sat	sat	sat	sat	sat
$a=2$		sat	sat	**unsat**	unsat	unsat	unsat
$a=3$			sat	sat	**unsat**	unsat	unsat
$a=4$				sat	sat	**sat**	**unknown**
$a=5$					sat	sat	**unknown**
$a=6$						sat	sat
$a=7$							sat

hyperedges connected (which can force unsatisfiability)" against "each hyperedge being incident on a large number of vertices (which can force satisfiability)" [1, §8.1]. They tabulate what they know, for $a, b \leq 7$, in a table captioned "A table showing the satisfiability statuses of complete uniform hypergraphs. Non-obvious results are shown in boldface." [1, Table 3], reproduced here as Table 1, and they record the gap in as many words [1, §8.1]:

> "As seen in the table, the satisfiability-status of $K(4,7)$ and $K(5,7)$ are not known."

That sentence is the question this paper answers, and the answer runs in both directions.

Theorem 1.1. $K(4,7)$ *is not totally satisfiable: there is a CNF living on it — one clause on each of the* 35 *four-element subsets of a seven-element vertex set — that no assignment satisfies. An explicit such CNF is given in Table 2.*

Theorem 1.2. $K(5,7)$ *is totally satisfiable.*

Theorem 1.2 is a two-line count, and we say so at once: each of the $\binom{7}{5} = 21$ clauses of a CNF living on $K(5,7)$ is falsified by exactly $2^{7-5} = 4$ of the 128 assignments, and $21 \cdot 4 = 84 < 128$ leaves an assignment satisfying all of them. The same argument proves the cell $K(4,6)$ that the same table marks boldface, so the tool was in reach; what is new here is the value of the cell, not the method. Theorem 1.1 is where the content of the pair sits: it is the one of the two cells in which the counting argument leaves room for a covering, so that a covering had to be found rather than excluded.

A correction. While recomputing the table we found that one cell of it, printed in boldface, is wrong.

Theorem 1.3. $K(3,5)$ *is totally satisfiable. Equivalently: no choice of a sign pattern for each of the ten three-element subsets of a five-element set yields an unsatisfiable CNF.*

Table 1 prints this cell as **unsat**. We state the correction with some care, because it is a claim against a specific document: the assertion we contradict is the cell $a = 3$, $b = 5$ of [1, Table 3], in the `v1` e-print of arXiv:2105.11390, submitted 24 May 2021. That is the only version; the arXiv record carried no later version and no journal reference when we checked it on 3 September 2026. Theorem 1.3 rests on an exhaustive search, described in §6, which was carried out four times by four differently-structured programs before it was carried out once more inside a proof assistant.

The threshold sequence. Total satisfiability is inherited by sub-hypergraphs ([1, Lemma 3.2]; Proposition 3.4 below), and $K(a,b)$ is a sub-hypergraph of $K(a,b')$ whenever $b \leq b'$. So for each $a \geq 2$ the status of $K(a,b)$ switches once, from `sat` to `unsat`, as b grows, and

$$t(a) \;:=\; \min\{\, b : K(a,b) \text{ is not totally satisfiable} \,\}$$

determines the whole row. The published table reaches $b = 7$, so it determines $t(2) = 4$, records $t(3)$ wrongly as 5, and leaves $t(4)$ and $t(5)$ undecided.

Theorem 1.4. *$t(a) = 4$, 6, 7, 8, 9, 11, 12 for $a = 2, 3, 4, 5, 6, 7, 8$ respectively.*

Set $m(a) := \min\{b : \binom{b}{a} \geq 2^a\}$. The counting argument behind Theorem 1.2 says exactly that $t(a) \geq m(a)$: a covering of the b-cube by $\binom{b}{a}$ sets of 2^{b-a} points each needs $\binom{b}{a}2^{b-a} \geq 2^b$. The interest of Theorem 1.4 is how tight that trivial bound turns out to be.

Theorem 1.5. *$t(a) = m(a)$ for $a = 2, 4, 5, 6, 7, 8$. At $a = 3$ the two differ: $m(3) = 5$, since $\binom{5}{3} = 10 \geq 8 = 2^3$, while $t(3) = 6$.*

So on the computed range the elementary bound is exact except at one value of a, and that one value is the cell of [1, Table 3] that is printed incorrectly. Both facts are visible in the same place, and neither is visible without the other: the reason $K(3, 5)$ is hard to get right by hand is exactly the reason $t(3) > m(3)$ is worth recording.

A search of the On-Line Encyclopedia of Integer Sequences [5] on 3 September 2026 for 4, 6, 7, 8, 9, 11, 12 returned ten sequences, all numerical coincidences with unrelated definitions, as did the search for 2, 4, 6, 7, 8, 9, 11, 12; a positive control on 1, 1, 2, 3, 5, 8, 13, 21 returned the Fibonacci numbers as expected. We take this as evidence that the threshold sequence has not been tabulated, not as evidence that it could not have been. The counting bound is another matter. The search for 2, 4, 5, 7, 8, 9, 11, 12, 13 returns `A254058`, "Smallest $a(n)$ such that $C(a(n), n) \geq 2^n$" [5] — which is m exactly, entered in the OEIS in January 2015, and whose 67 published terms agree with $m(1), \ldots, m(67)$ as we recompute them. So of the two sequences compared in Theorem 1.5 the elementary one is on record and the one it bounds is not.

What was searched. Because the results above are in part claims about what is *not* in the literature, we say what was looked at, and when. All of the following was carried out on 2 September 2026 and repeated on 3 September 2026. The arXiv record of [1] lists one version and no journal reference, and zbMATH and DBLP both hold the paper as a preprint only, so the two cells printed `unknown` were never revised. Forward citations, taken from OpenAlex, are a single unrelated 2024 workshop note on a reduction from 3-SAT to independent set for [1], and none at all for [2]. A search of arXiv for "GraphSAT" returns three papers, of which two are unrelated: arXiv:2408.14122, whose graph-neural-network classifier for encrypted network traffic is called GraphSAT, and arXiv:2105.12908, which refers to a SAT solver of that name ("solvers with ad hoc constraint propagators for acyclicity and reachability constraints such as GraphSAT"). The name is in use for at least two things besides the decision problem of [1], and neither is related to it. A sweep of a local full-text corpus of arXiv papers for *GraphSAT*, for the identifier 2105.11390 and for the phrase *totally satisfiable* produced, apart from [1] itself, only substring coincidences and two unrelated uses of the phrase, one for T-automata and one model-theoretic. The authors' software repository [3] contains no table of complete uniform hypergraphs, and no later paper in their arXiv or DBLP listings revisits it. The nearest neighbouring results we located are [4], which is asymptotic and decides no cell here, and the classical fact that an unsatisfiable k-CNF needs at least 2^k clauses, which is Proposition 3.3 in another guise. We did not search MathSciNet, Google Scholar or theses. The negative should be read as "not found", not as "new".

What is not settled. The first cell we cannot decide is $K(9, 13)$. It sits where the counting bound goes silent for $a = 9$: $\binom{12}{9} = 220 < 512 = 2^9 \leq 715 = \binom{13}{9}$, so $m(9) = 13$ and the bound gives $t(9) \geq 13$ (Proposition 9.1) but says nothing about $K(9, 13)$ itself. We stop there; §9 records what deciding it would cost. We also prove no general theorem: every statement here is about a named cell or a finite list of them, and the two uniform statements one would like — that $\binom{b}{a} < 2^a$ implies total satisfiability for *all* a, b, and a construction giving $t(a) = m(a)$ for all $a \neq 3$ in one stroke — are not proved.

Organisation. §2 fixes the definitions. §3 gives the covering reformulation on which everything rests, and the counting bound and monotonicity that come with it. §4 describes the exhaustive search and the two facts that make it complete. §5 proves Theorems 1.1 and 1.2, §6 Theorem 1.3, and §7 Theorems 1.4 and 1.5. §8 reports the reproduction of the published material and the negative controls, §9 the frontier, and §10 the machine verification.

2. Preliminaries

We write $[b] = \{1, \dots, b\}$ for the vertex set and use variables $x_1, \dots, x_b$. An *assignment* is a map x from variables to $\{0,1\}$; nothing below restricts attention to the first b variables, and the formal development quantifies over assignments defined on all of $\mathbb{N}$. A *literal* is x_v or $\neg x_v$, a *clause* a disjunction of literals, and a *CNF* a conjunction of clauses.

A *simple hypergraph* on $[b]$ is a finite family g of pairwise distinct non-empty subsets of $[b]$, its *edges*. (This is the multiplicity-one case of the looped-multi-hypergraphs of [1]; the whole paper stays inside it.)

Definition 2.1. Let g be a simple hypergraph on $[b]$. A *sign choice* for g is a family $f = (f_S)_{S \in g}$ with $f_S \in \{0,1\}^S$. It determines the clause

$$c_{S,f_S} = \bigvee_{v \in S,\ f_S(v)=0} x_v \ \vee \bigvee_{v \in S,\ f_S(v)=1} \neg x_v$$

on each edge, and the CNF $\Phi_{g,f} = \bigwedge_{S \in g} c_{S,f_S}$. The CNFs *living on* g are exactly the $\Phi_{g,f}$, one for each sign choice. We call g *totally satisfiable* if $\Phi_{g,f}$ is satisfiable for every sign choice f, and *unsatisfiable* otherwise.

Two remarks on the fit with [1]. First, for a simple hypergraph the set of CNFs living on g is never empty — with multiplicity n on an edge of size k there are $\binom{2^k}{n}$ clause choices, which vanishes only when $n > 2^k$ — so the nonemptiness clause of the quoted definition is automatic here, and Definition 2.1 is the quoted definition specialised, not weakened. It is, however, not vacuous by accident: the empty hypergraph is totally satisfiable under Definition 2.1, which is precisely why [1] requires a graph to be a non-empty multiset of edges. Second, we parametrise sign choices by the edge, as a function on g, rather than by position in a list of edges; the two agree because the edges of a simple hypergraph are pairwise distinct, and for the hypergraphs $K(a,b)$ used below this distinctness is checked rather than assumed (Proposition 8.3 (v)).

Definition 2.2. $K(a,b)$ is the simple hypergraph on $[b]$ whose edges are all $\binom{b}{a}$ subsets of size exactly a.

This is the reading forced by [1, Table 3] itself. The source's sentence introducing $K(a,b)$ adds "We can also think of this as the $(a-1)$-skeleton of a $(b-1)$-simplex", which taken literally would include all faces of size at most a; but with the lower faces present $K(2,3)$ would be unsatisfiable, its three singleton faces and one of its pairs carrying $x_1 \wedge x_2 \wedge x_3 \wedge (\neg x_1 \vee \neg x_2)$, and the table prints that cell `sat`. Edges of size exactly a is the reading we use throughout.

Finally, we write sign patterns as bit strings: for $S = \{v_1 < \cdots < v_a\}$ the pattern f_S is written $f_S(v_1) f_S(v_2) \cdots f_S(v_a)$. Thus the edge 1234 with pattern 0110 carries the clause $x_1 \vee \neg x_2 \vee \neg x_3 \vee x_4$.

3. The covering reformulation

Everything below is an instance of one elementary observation: a clause on S has exactly one falsifying pattern on S.

Definition 3.1. For $S \subseteq [b]$ and $p \in \{0,1\}^S$, the *cylinder* $\mathcal{C}(S,p)$ is $\{x \in \{0,1\}^b : x|_S = p\}$, a subcube of $\{0,1\}^b$ of codimension $|S|$ and size $2^{b-|S|}$.

Proposition 3.2 (Covering reformulation)**.** *Let g be a simple hypergraph on $[b]$ and f a sign choice for g. Then the assignment x falsifies the clause c_{S,f_S} if and only if $x \in \mathcal{C}(S, f_S)$. Consequently $\Phi_{g,f}$ is unsatisfiable if and only if*

$$\bigcup_{S \in g} \mathcal{C}(S, f_S) \;=\; \{0,1\}^b,$$

and g is totally satisfiable if and only if no sign choice for g covers the cube.

Proof. The clause c_{S,f_S} has one literal per $v \in S$, and the literal at v is falsified by x exactly when $x(v) = f_S(v)$; a disjunction is falsified exactly when every disjunct is. So x falsifies c_{S,f_S} iff $x|_S = f_S$. Hence x satisfies $\Phi_{g,f}$ iff x lies outside every cylinder, and $\Phi_{g,f}$ is unsatisfiable iff the cylinders exhaust $\{0,1\}^b$. The last sentence is the contrapositive, quantified over f. □

We record two immediate consequences, both of which we use as tools rather than state as results.

Proposition 3.3 (Counting bound)**.** *If $\binom{b}{a} \cdot 2^{b-a} < 2^b$, equivalently $\binom{b}{a} < 2^a$, then $K(a, b)$ is totally satisfiable. Hence $t(a) \geq m(a)$ with $m(a) = \min\{b : \binom{b}{a} \geq 2^a\}$.*

Proof. Each of the $\binom{b}{a}$ cylinders has 2^{b-a} points, so their union has fewer than 2^b points and cannot be all of $\{0,1\}^b$. Apply Proposition 3.2. □

Proposition 3.3 is the one statement in this paper that we prove on paper for general a and b but do not machine-check in that generality: in the formal development it is instantiated one cell at a time, as the depth-zero case of the search of §4. Every *use* of it below is therefore machine-checked; the uniform statement is not.

Proposition 3.4 (Monotonicity)**.** *Let $g \subseteq h$ be simple hypergraphs on $[b]$. If g is not totally satisfiable then neither is h; equivalently, if h is totally satisfiable then so is g. In particular $K(a, b)$ is totally satisfiable for every $b < t(a)$ and for no $b \geq t(a)$, so $t(a)$ determines the whole row a of the table.*

Proof. A sign choice covering the cube through the edges of g extends to h by choosing the remaining patterns arbitrarily; the union only grows. For the last sentence, all a-subsets of $[b]$ are a-subsets of $[b']$ when $b \leq b'$, so $K(a, b) \subseteq K(a, b')$ after the evident relabelling. □

Proposition 3.4 is not new: [1] states that total satisfiability is "closed under subgraphing" in its introduction and proves the corresponding equi-implication as its Lemma 3.2, and its §7.6 gives the compactness version, "an infinite graph is totally satisfiable if and only if every finite subgraph of it is totally satisfiable". We reprove it here only because the threshold $t(a)$ is meaningless without it.

4. The exhaustive search

To prove that a hypergraph *is* totally satisfiable when the counting bound is silent one must rule out every sign choice. For $K(3, 5)$ that is $8^{10} = 1\,073\,741\,824$ choices, and for larger cells it is hopeless without pruning. We use a depth-first search over the edges in a fixed order, carrying the list of points not yet covered, with exactly two devices.

The capacity cut. If r edges remain unassigned and each cylinder contains at most κ points, then at most κr further points can be covered; so a node with more than κr uncovered points is dead. Soundness of this cut, and completeness of the search, both come from one counting lemma: if every element of a list L satisfies at least one of the predicates $p_1, \dots, p_r$, then $|L| \leq \sum_i |\{y \in L : p_i(y)\}|$. Applied to the uncovered points and the cylinder-membership predicates of the remaining edges, it says that a covering extension forces $|L| \leq \kappa r$.

Normalisation by negation. For $t \in \{0,1\}^b$, negating the variables in the support of t maps coverings to coverings: if (f_S) covers, so does $(f_S \oplus t|_S)$, because $x \in \mathcal{C}(S, f_S)$ iff $x \oplus t \in \mathcal{C}(S, f_S \oplus t|_S)$. Choosing t to be (any extension of) the pattern on a fixed first edge S_0, we may assume $f_{S_0} = 0 \cdots 0$. This divides the search by 2^a at no cost.

TABLE 2. A sign choice on $K(4,7)$ with no satisfying assignment. Each entry gives a four-element subset $S \subseteq \{1,\dots,7\}$ and the pattern f_S, written in increasing order of the vertices of S; the clause is $\bigvee_{v\in S} \ell_v$ with $\ell_v = x_v$ when $f_S(v) = 0$ and $\ell_v = \neg x_v$ when $f_S(v) = 1$. The 35 cylinders $\mathcal{C}(S, f_S)$ cover all 128 points of $\{0,1\}^7$.

1234	0000	1256	0110	1357	0111	2346	1111	2467	0000
1235	1000	1257	0001	1367	0000	2347	1001	2567	0000
1236	0010	1267	1011	1456	1011	2356	1011	3456	1010
1237	1100	1345	1100	1457	1100	2357	1100	3457	0111
1245	0100	1346	1110	1467	0000	2367	0000	3467	0000
1246	0011	1347	0100	1567	0000	2456	0111	3567	0000
1247	0010	1356	1010	2345	1010	2457	1101	4567	0000

Proposition 4.1 (Completeness of the search)**.** *The depth-first search with the capacity cut is complete: if some sign choice covers the cube, the search finds one. It remains complete after the first edge's pattern is fixed to* $0\cdots 0$. *Consequently, a run that terminates without finding a covering is a proof that the hypergraph is totally satisfiable.*

The proof is a routine induction on the list of edges still to be assigned, using the counting lemma above at each node for the cut and the negation symmetry once at the root; it is machine-checked, and we do not reproduce it. Both halves of Proposition 4.1 matter for what follows: the `sat` results below are *negative* search results, and are worth no more than the completeness proof behind them.

We use the search in two regimes. When $\binom{b}{a}2^{b-a} < 2^b$ the capacity cut fires at the root — the root has 2^b uncovered points and $\binom{b}{a}$ edges of capacity 2^{b-a} — so the counting bound of Proposition 3.3 is not a separate argument but the depth-zero case of the same search, and this is how it is machine-checked, cell by cell. When the cut does not fire at the root the search runs to completion; among the totally satisfiable hypergraphs treated below this happens at $K(1,b)$ for $b \geq 2$, where the search closes almost at once, and at $K(3,5)$, where it does not. On the `unsat` side no search is needed for the final certificate: an explicit sign choice, checked against all 2^b points, is a complete proof by Proposition 3.2.

5. The two cells left open

Proof of Theorem 1.1. Table 2 lists a pattern $f_S \in \{0,1\}^S$ for each of the 35 four-element subsets S of [7]. A direct check over the $2^7 = 128$ assignments shows that every one of them lies in at least one of the 35 cylinders $\mathcal{C}(S, f_S)$; by Proposition 3.2 the corresponding 35-clause CNF is unsatisfiable, and $K(4,7)$ is not totally satisfiable. □

The proof is a computation over 128 points and 35 cylinders and can be redone by hand in an afternoon or by machine in microseconds; the work was in finding the witness. The counting bound gives no help here — $\binom{7}{4}\cdot 2^3 = 280 > 128$, so there is more than twice the capacity needed — but capacity is not arrangement: a sign choice drawn uniformly at random leaves $128\,(1-2^{-4})^{35} \approx 13$ of the 128 points uncovered in expectation, so the witness had to be searched for. The witness of Table 2 was produced by a depth-first search that, at each node, takes the smallest uncovered point X and branches over which still-unassigned edge S is to cover it, the pattern being then forced to $X|_S$; it reached a covering after a few dozen nodes. The table was then re-verified by an independently written program, and finally checked inside a proof assistant.

Proof of Theorem 1.2. $\binom{7}{5}\cdot 2^{7-5} = 21\cdot 4 = 84 < 128 = 2^7$, so Proposition 3.3 applies. □

Remark 5.1. The same one-line argument settles the boldface cell $K(4,6)$ of Table 1, as $15\cdot 4 = 60 < 64$. Read the other way, it locates the whole difficulty of the table. Of the 28 cells of Table 1, 14 satisfy $\binom{b}{a} < 2^a$ and are `sat` by Proposition 3.3 alone; in the other 14 — the row $a = 1$ from $b = 2$

on, the cells $K(2,b)$ with $b \geq 4$, the cells $K(3,b)$ with $b \geq 5$, and $K(4,7)$ — the count leaves room for a covering. Every one of the latter is `unsat` except the trivial row $a = 1$, where a CNF living on $K(1,b)$ is a conjunction of b unit clauses on b distinct variables and hence satisfiable, and the single cell $K(3,5)$. So within the printed range $K(3,5)$ is the only cell where the count permits a covering, the answer is nevertheless `sat`, and the reason is not trivial. We note this not to diminish [1] but to say precisely where its two unknown cells sat: one was within reach of a tool its authors had already used, and the other was not.

6. The correction at $K(3,5)$

$K(3,5)$ is the one cell of Table 1 where the counting bound is silent and the answer is nevertheless `sat`: $\binom{5}{3} \cdot 2^2 = 40 > 32$, so ten cylinders of four points each have more than enough room to cover the 32-point cube, and yet no arrangement of them does.

Proof of Theorem 1.3. By Proposition 3.2 it suffices to show that no choice of $f_S \in \{0,1\}^S$, one for each of the ten triples $S \subseteq [5]$, has $\bigcup_S \mathcal{C}(S, f_S) = \{0,1\}^5$. This is a finite statement about $8^{10} = 1\,073\,741\,824$ choices, and we settle it by exhaustive search. Fixing the edges in a lexicographic order and using the capacity cut of §4, the search tree has 28 707 401 nodes and contains no covering. Normalising the pattern of the first triple to 000 by the negation symmetry (Proposition 4.1) reduces this to 3 588 426 nodes, again with no covering; this is the run carried out inside the proof assistant. □

This is the one claim in the paper that rests on a search large enough to be worth double-checking, so we say exactly what was run. Four programs, written separately and sharing no code, agree that no covering exists:

(1) the fixed-edge-order enumeration with the capacity cut, 28 707 401 nodes;
(2) the same with the first triple's pattern normalised to 000, 3 588 426 nodes — this is the run reproduced in the proof assistant;
(3) a smallest-uncovered-point depth-first search with exact-cover-style exclusion, 410 056 nodes;
(4) an unpruned smallest-uncovered-point depth-first search over a bitset of points, in a separate program, 9 864 101 nodes.

The four differ in branching rule, in pruning and in data structure, so a shared bug would have to be a shared misreading of the problem rather than a shared coding error; against that, the same programs reproduce every other cell of Table 1 and the ten named-graph claims of [1] discussed in §8. The node counts of (1) and (2) are determined by the description above — fixed lexicographic edge order, capacity cut, one node per visited partial choice — and a reader who reimplements it will get those two numbers; the counts of (3) and (4) depend on details we do not fix, and are quoted only to show that the four runs were genuinely different.

Remark 6.1 (A second slip in the source, this one textual). The paragraph of [1, §8.1] that points at Table 1 opens with the sentence

> "For example, we know that $K(2,4)$ is K_4, i.e. the complete simple graph on 4 vertices and is known to be totally satisfiable. On the other hand, the graph $K(2,3)$ is C_3 is known to be totally satisfiable."

Here the table is right and the sentence is wrong: K_4 is one of the four obstructions in the theorem of [2] that the same paper quotes in its introduction, the table prints the cell $a = 2$, $b = 4$ as boldface `unsat`, and $K(2,4)$ is indeed not totally satisfiable (Theorem 8.1). The "On the other hand" shows that a contrast was intended. This is a slip of the pen and not a mathematical error, and we record it only because it stands in the very paragraph that points at the cell which *is* a mathematical error, and could otherwise be confused with it.

7. The threshold sequence

Proof of Theorem 1.4. By Proposition 3.4 it suffices, for each a, to prove one `sat` and one `unsat` statement:

a :	2	3	4	5	6	7	8
totally satisfiable:	$K(2,3)$	$K(3,5)$	$K(4,6)$	$K(5,7)$	$K(6,8)$	$K(7,10)$	$K(8,11)$
not totally satisfiable:	$K(2,4)$	$K(3,6)$	$K(4,7)$	$K(5,8)$	$K(6,9)$	$K(7,11)$	$K(8,12)$

Every entry of the first row except $K(3,5)$ is an instance of the counting bound, Proposition 3.3: the products $\binom{b}{a}2^{b-a}$ are $3 \cdot 2 = 6 < 8$, $15 \cdot 4 = 60 < 64$, $21 \cdot 4 = 84 < 128$, $28 \cdot 4 = 112 < 256$, $120 \cdot 8 = 960 < 1024$ and $165 \cdot 8 = 1320 < 2048$ against cubes of 2^b points. The entry $K(3,5)$ is Theorem 1.3. Every entry of the second row is an explicit sign choice whose cylinders cover the cube, verified point by point as in the proof of Theorem 1.1; the choices have 6, 20, 35, 56, 84, 330 and 495 clauses respectively, and $K(4,7)$ is Theorem 1.1. $\square$

The seven covering witnesses were found by a randomised sequential greedy procedure — fix a random order of the edges, give each edge in turn the pattern covering the most currently-uncovered points, and restart from a new random order on failure. How many restarts each cell took we do not report: the counts depend on the random stream and are not reproducible from that description. What is reproducible is which cells are tight. The capacity a covering has to play with, $\binom{b}{a}2^{b-a}/2^b = \binom{b}{a}2^{-a}$, is 1.31 at $K(6,9)$ and 1.5 at $K(2,4)$ against 1.75 to 2.58 at the other five cells — and those two were exactly the two on which the greedy procedure needed more than one restart. Each witness was re-verified by an independent program before being written into the formal development, and each is verified there against all 2^b points. We stress that the greedy procedure is a search heuristic and carries no part of the proof: the certificate is the witness, and checking that a witness is a covering is a finite computation with no heuristics in it.

Proof of Theorem 1.5. The counting bound values are read off $\binom{b}{a}$ against 2^a:

a	2	3	4	5	6	7	8
2^a	4	8	16	32	64	128	256
$\binom{m(a)-1}{a}$	3	4	15	21	28	120	165
$\binom{m(a)}{a}$	6	10	35	56	84	330	495
$m(a)$	4	5	7	8	9	11	12
$t(a)$	4	6	7	8	9	11	12

and compared with Theorem 1.4. They agree except at $a = 3$, where $\binom{5}{3} = 10 \geq 8$ gives $m(3) = 5$ while $t(3) = 6$ by Theorem 1.3. $\square$

Remark 7.1. The gap at $a = 3$ is a genuine failure of the converse to Proposition 3.3, and not an artefact of the range: "$\binom{b}{a} \geq 2^a$ implies $K(a,b)$ is not totally satisfiable" is false, with $(a,b) = (3,5)$ the witness (Proposition 8.3 (iii)). Whether $a = 3$ is the only value of a at which $t(a) > m(a)$ we do not know; $a = 9$ is the first value at which the question is open (§9).

8. Reproduction of the published material, and controls

Before claiming anything new we recomputed everything the source asserts that our model can express. Two statements record this.

Theorem 8.1. *Every cell of Table 1 that carries a status — that is, the 26 of its 28 cells that are not marked* `unknown` *— has the status printed there, with the single exception of $a = 3$, $b = 5$. That cell is corrected by Theorem 1.3, and the two cells marked* `unknown` *are settled by Theorems 1.1 and 1.2.*

Theorem 8.2. *The ten satisfiability statuses that* [1] *asserts, or immediately implies, for explicitly named graphs all hold. Not totally satisfiable:* K_4*, the butterfly, the book* B_3 *and the bowtie — the four obstructions of* [2, Theorem 18 and Remark 19]*, quoted as an unnumbered theorem in* [1, §1]*, with* K_4 *called "a known unsatisfiable graph" again in* [1, §6.2]*. Totally satisfiable:* $K_4 - e$ *and the cycle* C_4*, neither of which contains an obstruction as a topological minor; the cycle* $C_3 = K(2,3)$*; the face set* $\{abc, acd, abd, bcd\} = K(3,4)$ *of the tetrahedron, of which* [1, §6.2] *says "Thus the tetrahedron is totally satisfiable"; and the two minimal triangulations of the triangular prism,* $123\,|\,125\,|\,134\,|\,145\,|\,236\,|\,256\,|\,346\,|\,456$ *and* $123\,|\,125\,|\,136\,|\,145\,|\,146\,|\,236\,|\,256\,|\,456$*, of which* [1, §6.2] *says "both triangulations are totally satisfiable".*

Theorem 8.2 is a reproduction, not a result, and it is the gate this work had to pass before Theorem 1.3 could be believed: a model in which $K(3,5)$ came out totally satisfiable but the source's own graphs did not is a model of something else. The ten split three ways: four are the obstructions of the quoted theorem of [2], four are asserted in so many words in the text of [1], and the statuses of $K_4 - e$ and C_4 follow from the "only if" half of that same quoted theorem rather than from a sentence. Those last two carry more weight than the rest: they are its minimality half, and a model that merely reported "dense implies unsatisfiable" would get them wrong. The four obstruction graphs were transcribed from the drawing macros of the `v1` e-print rather than read off a rendered figure, and the prism triangulations from the displayed formulas quoted above.

Proposition 8.3 (Controls)**.** *The following hold, and each refutes a natural over- or under-reading of the results above.*

(i) *It is not the case that* $K(3,b)$ *is totally satisfiable for every* b*:* $K(3,6)$ *is not. So Theorem 1.3 is not an instance of a blanket statement about* $a = 3$*.*
(ii) *It is not the case that* $b > a$ *forces failure:* $K(4,6)$ *is totally satisfiable.*
(iii) $\binom{b}{a} \geq 2^a$ *does not imply that* $K(a,b)$ *fails to be totally satisfiable;* $(a,b) = (3,5)$ *is a counterexample.*
(iv) $K(4,7)$ *also carries satisfiable CNFs, so Theorem 1.1 says that* some *CNF living on it is unsatisfiable, not that all are; and the empty hypergraph is vacuously totally satisfiable.*
(v) *For* $(a,b) = (2,4), (3,5), (4,7), (5,7)$*, the edge list of* $K(a,b)$ *consists of exactly* $\binom{b}{a}$ *pairwise distinct lists of* a *distinct vertices of* $[b]$*.*

Clause (v) is the structural control. Without it every statement of the form "$K(a,b)$ is totally satisfiable" proved here could be a statement about a different hypergraph; with it, the object named in Definition 2.2 is the object the theorems are about.

9. The frontier

The cell where this paper stops is $K(9,13)$. The counting bound settles the row $a = 9$ up to $b = 12$ and goes silent at $b = 13$: $\binom{12}{9} = 220 < 512 = 2^9 \leq 715 = \binom{13}{9}$. Among the cells $K(a, m(a))$ — those where the count first permits a covering — it is, after $K(3,5)$, the second that does not yield one to the greedy search of §7; unlike $K(3,5)$, an exhaustive settlement of it is out of reach here.

Proposition 9.1. $t(9) \geq 13$*; that is,* $K(9,12)$ *is totally satisfiable, by the counting bound* $220 \cdot 2^3 = 1760 < 4096 = 2^{12}$*.*

Remark 9.2 (Measurements, outside the formal development). Nothing in this paper depends on the following, and none of it is machine-checked; it is recorded so that the next attempt on $K(9,13)$ starts from a price rather than from optimism. The randomised greedy procedure of §7, run for 200 restarts, leaves at best 49 of the 8192 points of $\{0,1\}^{13}$ uncovered; adding a local-search phase that re-chooses one edge's pattern at a time to a local optimum, 40 restarts leave 46. Both are single runs and depend on the random stream, and we give no timings with them: they were made on a shared machine under load, where a wall-clock number means nothing to anyone else. What is exact is the size of the natural SAT encoding. It needs one auxiliary variable per (edge, point) pair, $715 \cdot 8192 = 5\,857\,280$ of them, with nine binary clauses each: about 52.7 million clauses of width 2 together with 8192 clauses of width 715, roughly 2 gigabytes of DIMACS before any solving begins. We did not run it.

Two further directions are worth naming. First, the results here are per-cell, and the two uniform statements that would replace the table by a theorem are open in this development: that $\binom{b}{a} < 2^a$ implies total satisfiability for all a, b (proved here one cell at a time, because the cylinder-size fact is checked by enumeration rather than proved in general), and a construction giving a covering at $b = m(a)$ for every $a \neq 3$ in one stroke. Second, Theorem 1.3 deserves a human proof, and a partial one exists: splitting the 5-cube at a vertex v and writing $k_0(v)$ for the number of triples through v whose pattern sends v to 0, a capacity count gives $2 \leq k_0(v) \leq 4$ and an exact-partition argument excludes $k_0(v) = 2$ and hence $k_0(v) = 4$, forcing $k_0(v) = 3$ at every vertex; the remaining case is unfinished, and finishing it would replace the one large computation in this paper by an argument.

10. Verification

Formalised and machine-checked in Lean 4 [6] are Propositions 3.2, 3.4, 4.1, 8.3 and 9.1, Theorems 1.1–1.5, and Theorems 8.1 and 8.2. Not formalised are Proposition 3.3 in its general form — it is machine-checked once per cell instead, as explained where it is stated — and the measurements of Remark 9.2, which are labelled as measurements there and on which nothing depends. The development imports no mathematical library: literals, clauses, assignments as total maps $\mathbb{N} \to \{0, 1\}$, the coding of an assignment as a natural number and the cylinder arithmetic as bit masks are all elementary list and integer operations, and the passage from arbitrary total assignments to the 2^b bit-coded ones is proved rather than assumed, so that no statement is silently restricted to assignments on the first b variables. It declares no axiom of its own and contains no `sorry`. Nine declarations delegate a finite evaluation to Lean's compiled evaluator (`native_decide`), contributing sixteen evaluation axioms in all: the $3\,588\,426$-node search of Theorem 1.3, one axiom; the six cells $K(6, 8)$, $K(6, 9)$, $K(7, 10)$, $K(7, 11)$, $K(8, 11)$ and $K(8, 12)$ that carry Theorem 1.4 for $a \geq 6$, two each; and the two evaluations behind Proposition 9.1, three between them. Those nine are the only trusted computations in the paper, and the statements that inherit an evaluation axiom are exactly the ones that quote them: Theorem 1.3, clause (iii) of Proposition 8.3, the row $a = 3$ of Theorem 1.4 and hence Theorem 1.4 as a whole, and the entries with $a \geq 6$ in Theorems 1.4 and 1.5. Everything else is checked by the Lean kernel alone: Propositions 3.2 and 4.1 and the rest of the machinery of §3–§4, both implications of Proposition 3.4, Theorems 1.1 and 1.2, Theorem 8.1 in the form it is checked — the twenty-seven cells other than $K(3, 5)$, both formerly unknown cells included — Theorem 8.2, the other four clauses of Proposition 8.3, the arithmetic of Theorem 1.5, and both halves of every row of Theorem 1.4 with $a \leq 5$ except the cell $K(3, 5)$, the witness check for $K(5, 8)$ included. The reason for the split is throughput and nothing else: on the same cylinder-membership test the kernel evaluates on the order of 10^3 per CPU second and the compiled evaluator $3\text{–}4 \cdot 10^6$, and the checks above $a = 5$ run to 10^8 tests and more. Rechecking the whole development costs about 271 seconds of CPU time. Independently written programs, sharing no code with the formal development, reproduce every covering witness and every search verdict reported above. The repository is private; repository reference to be added at submission.

References

[1] V. Karve and A. N. Hirani, *GraphSAT — a decision problem connecting satisfiability and graph theory*, arXiv:2105.11390v1 [math.CO] (cross-listed cs.DM, math.LO), submitted 24 May 2021. All section, lemma and table numbers cited above are those of the `v1` e-print, which is the only version: as of 3 September 2026 the arXiv record lists no later version, no DOI and no journal reference, zbMATH holds the item as "Preprint, arXiv:2105.11390 [math.CO] (2021)" with no source journal, and DBLP lists it only as a CoRR entry.

[2] V. Karve and A. N. Hirani, *The complete set of minimal simple graphs that support unsatisfiable 2-CNFs*, Discrete Appl. Math. **283** (2020), 123–132; `doi:10.1016/j.dam.2019.12.017`; arXiv:1812.10849. Volume, pages and DOI

checked against Crossref and zbMATH, 3 September 2026. Theorem 18 and Remark 19 are quoted, as a single unnumbered theorem, in the introduction of [1].

[3] V. Karve and A. N. Hirani, *vaibhavkarve/graphsat: graphsat v0.1.0* (software), Zenodo, 4 April 2021; `doi:10.5281/zenodo.4662169`. This is the release the bibliography of [1] cites; the DOI is the version DOI of `v0.1.0`, the concept DOI being `10.5281/zenodo.4662168`. Source repository `https://github.com/vaibhavkarve/graphsat`, last pushed 30 May 2023 (checked 3 September 2026); it contains no table of complete uniform hypergraphs.

[4] K. Chandrasekaran, N. Goyal and B. Haeupler, *Satisfiability thresholds for k-CNF formula with bounded variable intersections*, arXiv:1006.3030 [cs.DM], submitted 15 June 2010. The nearest neighbouring result we located: it is asymptotic and concerns families of k-CNFs whose clauses pairwise share few variables, and it decides none of the cells computed here.

[5] OEIS Foundation Inc., *The On-Line Encyclopedia of Integer Sequences*, `https://oeis.org`. Searched 3 September 2026 for the sequences quoted in §1; sequence `A254058`, `https://oeis.org/A254058`, is the counting bound m.

[6] L. de Moura and S. Ullrich, *The Lean 4 theorem prover and programming language*, in: A. Platzer and G. Sutcliffe (eds.), Automated Deduction — CADE 28, Lecture Notes in Computer Science **12699**, Springer, Cham, 2021, pp. 625–635; `doi:10.1007/978-3-030-79876-5_37`.