Hybrid Exact Search for Optimum Clique Coverings

Petr Kovář[1], Yifan Zhang[1,2,3,*]

[1]Department of Applied Mathematics, VSB–Technical University of Ostrava, Ostrava, Czech Republic
[2]Department of Mathematics, University of Ostrava, Ostrava, Czech Republic
[3]Department of Algebra, Charles University, Prague, Czech Republic

Abstract

We study optimum clique coverings of complete graphs using a combination of exact-search methods. The computation is split into three parts: finding decompositions, excluding smaller block counts, and checking the solver output. Dancing links (DLX) and integer linear programming (ILP) are used to find exact coverings with prescribed block counts. Boolean satisfiability (SAT) is used only after arithmetic, symmetry, and local-structure reductions have produced smaller infeasibility instances. All claimed decompositions are checked again by direct edge counting. We obtain $C^{\xi}(18, \{3, 4\}, 2) = 33$ and $C^{\xi}(19, \{3, 4\}, 2) = 35$ for exact coverings by triangles and quadruples. We also use the same method to prove the lower bound $C^{\xi}(33, \{3, 4, 5\}, 2) \geq 59$. Which of DLX and ILP is faster depends on the reduced instance, so both are retained as construction methods. The K_{33} computation shows how SAT can be used after several layers of filtering.

Keywords

clique covering, graph decomposition, exact cover, dancing links, integer linear programming, Boolean satisfiability, computational discrete mathematics

2020 MSC: 05B05, 05C70, 05C85, 90C10

1. Introduction

Optimum clique decompositions and coverings of complete graphs lie at the boundary between design theory and exact computation. Fix a set S of allowed clique orders. A covering of K_v is a multiset $\mathcal{B}$ of cliques of orders in S such that every edge of K_v is contained in at least one member of $\mathcal{B}$; each such clique is called a *block*. If m_e is the multiplicity with which an edge e is covered, the excess is $\xi(\mathcal{B}) = \sum_e (m_e - 1)$. We write $C^{\xi}(v, S, 2)$ for the minimum number of blocks among coverings with minimum possible excess. In the cases treated here the minimum excess is zero, so the problem is an exact decomposition problem. Equivalently, such a decomposition is a 2-$(v, S, 1)$ pairwise balanced design; see, for example, [4] for general background. For $S = \{3, 4\}$, if α and β are the numbers of blocks of orders 3 and 4, respectively, then (α, β) is the *block vector*; for $S = \{3, 4, 5\}$ we analogously use (α, β, γ).

General design-theoretic notation and the minimum-excess bounds used below are developed in [1, Sections 1–3]. An earlier construction-oriented use of dancing links for complete-graph coverings appears in [8]. Full treatments of the K_{18} and K_{19} reductions and of the layered K_{33} exclusion are available in [2, 3], respectively. The present paper summarizes the common exact-search procedure and points to the archived certificates and detailed case analyses supporting the three stated results.

The point of the paper is not to compare solvers on a list of unrelated examples. We use the examples to show how an exact proof can be organised when construction and exclusion have different computational forms. A decomposition may be found quickly by an exact-cover search, while a lower-bound proof usually requires many infeasibility checks. A direct ILP, DLX, or SAT model can always be written down, but in hard unsatisfiable cases such models may be too large or too symmetric to be useful. We therefore use the solvers at different stages of the proof: ILP and DLX for constructions, SAT for reduced exclusion problems, and an independent check at the level of edges.

The contributions are:

ITAT'26: Information Technologies – Applications and Theory, September 25–29, 2026, Biele Karpaty, Slovakia
*Corresponding author.
petr.kovar@vsb.cz (P. Kovář); yifan.zhang@vsb.cz (Y. Zhang)
© 2026 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

CEUR Workshop Proceedings ceur-ws.org ISSN 1613-0073
published 2026-11-09

CEUR-WS.org/Vol-4323/paper31.pdf

1. a uniform exact-search method for clique-covering problems, combining arithmetic reductions, canonical branch generation, ILP/DLX construction, SAT exclusions, and independent edge-level checks;
2. the exact covering numbers 33 for K_{18} and 35 for K_{19} with allowed orders $\{3,4\}$, which show the full construction-versus-exclusion procedure on medium-size instances;
3. the lower bound $C^{\xi}(33,\{3,4,5\},2) \geq 59$, where the K_{33} computation serves as a harder test of the same method and illustrates the use of layered SAT-assisted filtering;
4. scalability data showing why both DLX and ILP are kept as construction methods and why SAT is used only after reductions.

Thus the three computations have the same purpose in the paper: they demonstrate different parts of one exact-search procedure, rather than forming separate computational reports.

2. Hybrid exact search

Let $\mathcal{R}$ be the family of candidate cliques. The direct ILP formulation uses one binary variable x_B for every candidate block $B \in \mathcal{R}$ and the constraints

$$\sum_{B\in\mathcal{R}:\ e\subseteq B} x_B = 1 \qquad (e \in E(K_v)). \tag{1}$$

Prescribed block counts are added by equations such as $\sum_{|B|=3} x_B = \alpha$. This model is flexible and easy to audit, but it can be too broad for large symmetric infeasibility problems.

DLX represents the same exact-cover condition by rows and columns: rows are candidate blocks and columns are edges. It is a constructive backtracking engine with very fast reversible updates [5]. SAT uses Boolean variables for candidate blocks or for reduced local choices, together with exact-one or cardinality constraints in CNF; modern CDCL solvers such as Kissat are effective once the branch has already been reduced [6, 7].

The search is organised as follows.

1. **Arithmetic reduction.** Edge counts and local degree congruences restrict the possible block vectors.
2. **Canonical local reduction.** Triangle supports, local profiles, or distinguished skeletons are enumerated up to the relevant automorphism group.
3. **Construction.** DLX or ILP searches for a covering with the prescribed block counts.
4. **Lower-bound search.** SAT is used only on residual infeasibility instances after the reductions have fixed the relevant support or local signature data.
5. **Check.** Every decoded assignment or selected clique list is checked again for exact edge coverage and prescribed block counts.

Both ILP and DLX are kept for the construction step. The experiments below show that their relative strength depends on the reduced instance, so we do not choose one of them in advance.

3. The cases K_{18} and K_{19}

We first consider exact coverings of K_v by K_3's and K_4's for $v = 18, 19$. Let α and β denote the numbers of triangles and quadruples. The edge count is

$$3\alpha + 6\beta = \binom{v}{2}. \tag{2}$$

Thus minimizing $\alpha + \beta$ is equivalent to minimizing α. The general minimum-excess theory gives zero excess for $v = 18$ and $v = 19$ [1], so in both cases it remains to solve the exact decomposition problem.

Theorem 1. *The optimum exact covering numbers are*

$$C^{\xi}(18, \{3, 4\}, 2) = 33, \qquad C^{\xi}(19, \{3, 4\}, 2) = 35.$$

More precisely, K_{18} has an exact covering with $(\alpha, \beta) = (15, 18)$, and K_{19} has an exact covering with $(\alpha, \beta) = (13, 22)$; no smaller block count is possible.

Proof sketch. For $v = 18$, equation (2) gives $\alpha + \beta = (51 + \alpha)/2$. A covering with fewer than 33 blocks would have $\alpha \in \{7, 9, 11, 13\}$. For $v = 19$, similarly $\alpha + \beta = (57 + \alpha)/2$, so a covering with fewer than 35 blocks would have $\alpha \in \{7, 9, 11\}$. These lists also use the elementary local congruence lower bound $\alpha \geq 7$. The upper bounds are constructive and the lower bounds are reduced SAT exclusions. In the K_{18} case, DLX found the target $(15, 18)$-decomposition in 0.675 seconds, while an ILP feasibility run also verified a covering in 29.062 seconds. In the K_{19} case, the roles were reversed: ILP found the target $(13, 22)$-decomposition in 17.80 seconds, while the companion DLX run timed out at 86400 seconds under the same star-branch protocol. Thus the construction method was chosen from the reduced instance rather than fixed in advance. For the lower bound, SAT excludes all smaller admissible α-values; the K_{18} computation generated 21 residual CNFs, each of which was UNSAT, and the K_{19} computation included an exhaustive sweep over the 266 support cases generated by the reduction. The structural reductions and SAT encodings are described in detail in [2], and the construction and SAT archives contain the checkable certificates [11, 12, 13]. □

Table 1
Computation for the K_{18} and K_{19} optima.

instance	block counts	blocks	construction	lower-bound exclusion
K_{18} by $\{K_3, K_4\}$	$(15, 18)$	33	DLX 0.675 s; ILP 29.062 s, both verified	SAT excludes $\alpha = 7, 9, 11, 13$ by 21 residual UNSAT instances
K_{19} by $\{K_3, K_4\}$	$(13, 22)$	35	ILP 17.80 s verified; DLX timeout at 86400 s	SAT excludes $\alpha = 7, 9$, and all 266 support cases for $\alpha = 11$

Table 1 also explains why both ILP and DLX are used. The faster construction method changes between two closely related exact-decomposition instances.

4. Scalability checks and the direct K_{33} benchmark

Table 2 records a pure exact-cover benchmark: decomposing admissible complete graphs into triangles. In this sparse construction task, DLX is consistently fastest, while SAT and ILP have larger encoding overheads. The timings are wall-clock seconds from archived runs and are used as qualitative scalability evidence, not as hardware-independent performance claims [9, 10].

Table 3 gives construction data with prescribed block counts. These instances are closer to the construction step used for K_{18} and K_{19}: the block counts are prescribed, a first-vertex star branch is fixed, and the solver stops after the first verified exact covering. The label `star_a_b` denotes a branch in which the fixed first vertex lies in a triangular blocks and b quadruple blocks. DLX remains strong on many smaller cases, but ILP is faster on several prescribed-count branches.

We now turn to the hard mixed-clique instance on K_{33}. A general lower bound in [1, Theorem 8.1], applied to $v = 33$, gives

$$C^{\xi}(33, \{3, 4, 5\}, 2) \geq 57,$$

so block counts smaller than 57 are already excluded. At the 57-block threshold, the edge count leaves only two block vectors: $(2, 7, 48)$ and $(6, 0, 51)$. The first is excluded by [1, Lemma 7.2]; the remaining obstruction is $(6, 0, 51)$. Even after this arithmetic reduction, the candidate family contains

$$\binom{33}{3} + \binom{33}{5} = 5456 + 237336 = 242792$$

Table 2
Ten-point excerpt: decomposing K_v into copies of K_3.

v	ILP (s)	DLX (s)	SAT (s)
3	0.291475	0.000003	0.000014
15	0.015772	0.000038	0.004817
25	0.083603	0.000158	0.036953
37	0.475979	0.000639	0.082653
45	1.342822	0.001293	0.198544
57	5.207664	0.002766	0.286024
67	11.649263	0.005076	0.990463
79	30.630955	0.009297	0.724956
87	52.715849	0.013271	1.301034
99	115.984236	0.023774	35.759991

Table 3
Auxiliary $\{K_3, K_4\}$-construction experiments with prescribed block counts.

v	α	β	fastest construction	branch	time (s)	lower bound
7	7	0	DLX	`star_3_0`	7.00×10^{-5}	theorem
9	12	0	DLX	`star_4_0`	2.50×10^{-4}	theorem
10	9	3	DLX	`star_3_1`	6.00×10^{-4}	theorem
12	4	9	DLX	`star_1_3`	1.20×10^{-3}	theorem
13	0	13	DLX	`star_0_4`	1.10×10^{-3}	theorem
15	5	15	DLX	`star_1_4`	4.45×10^{-2}	theorem
16	0	20	ILP	`star_0_5`	8.28×10^{-3}	theorem
18	15	18	DLX	`star_1_5`	6.75×10^{-1}	SAT
19	13	22	ILP	`star_3_4`	1.78×10^{1}	SAT
21	8	31	ILP	`star_1_6`	1.22×10^{3}	theorem
22	7	35	ILP	`star_3_5`	1.03×10^{1}	theorem

Table 4
Direct one-hour benchmarks for the reduced K_{33} case $(\alpha, \beta, \gamma) = (6, 0, 51)$.

encoding	model size	cutoff	outcome
ILP	242,792 binary variables, 563 constraints, 3,835,568 nonzeros	3600 s	timeout, no feasible incumbent
DLX	242,792 candidate rows, 528 primary edge columns	3600 s	timeout, no exact cover
SAT	2,773,842 CNF variables and 7,593,150 clauses	3600 s	timeout, no SAT/UNSAT certificate

possible blocks. Direct one-hour runs on this reduced instance all timed out, as shown in Table 4 [15].

Tables 2–4 explain the choice of solvers. DLX is fast on sparse exact-cover searches. ILP can be faster when block counts and global side constraints are important. SAT is used after the lower-bound instance has been reduced to small residual feasibility questions.

5. The K_{33} lower-bound case

Assume that K_{33} is decomposed into six K_3's and fifty-one K_5's. If α_x and γ_x are the numbers of triangles and quintuples through a vertex x, then

$$32 = 2\alpha_x + 4\gamma_x,$$

so α_x is even. Moreover, $\sum_x \alpha_x = 3 \cdot 6 = 18$. Suppose that $\alpha_x \geq 4$ for some vertex x. The triangles through x contain $2\alpha_x$ distinct other vertices, and each of these vertices has positive, hence at least

two, triangle incidences. Consequently,

$$\sum_y \alpha_y \geq \alpha_x + 2(2\alpha_x) = 5\alpha_x \geq 20,$$

a contradiction. Thus every nonzero α_x equals 2, and exactly nine vertices are incident with the six triangles.

Form an auxiliary graph G whose six vertices correspond to the six triangles, with two vertices adjacent when the corresponding triangles meet. Each of the nine distinguished vertices lies in exactly two triangles and therefore determines an edge of G. Exactness prevents two triangles from sharing more than one vertex, so G is a simple cubic graph on six vertices. Its complement is a 2-regular graph on six vertices, hence is either C_6 or $2C_3$. Therefore G is isomorphic to the triangular prism or to $K_{3,3}$, and, after relabelling, the six triangles have one of the following two patterns on a distinguished K_9:

$$\text{grid: } 123, 456, 789, 147, 258, 369,$$
$$\text{prism: } 137, 128, 239, 467, 458, 569.$$

Let T be the union of these six triangles. In both patterns the graph $K_9 - T$ has clique number 3, so no quintuple contains four vertices from the distinguished K_9. A *type-s quintuple* is a K_5 meeting $V(T)$ in exactly s vertices. We call its K_9-part a *type-3 triple* when $s = 3$, and a *type-2 pair* when $s = 2$. Let n_s be the number of type-s quintuples. Then

$$(n_0, n_1, n_2, n_3) = (6 - n_3, 27 + 3n_3, 18 - 3n_3, n_3), \tag{3}$$

and a local count over the 24 outside vertices gives $n_3 \leq 4$. Thus there are exactly ten global cases: two triangle patterns and five values $n_3 = 0, 1, 2, 3, 4$.

Each case is treated by the same sequence of reductions:

1. fix the grid or prism triangle pattern and quotient by its automorphism group;
2. enumerate admissible type-3 triples and type-2 pairs on the distinguished K_9;
3. enumerate local signatures of outside vertices;
4. assemble global distributions consistent with (3);
5. apply a type-0 geometry filter for quintuples entirely outside the distinguished K_9;
6. run SAT on the surviving signature-assignment branches;
7. decode every SAT model and audit exact edge-disjointness and type-1 completion.

Tables 5 and 6 give the numerical summary. The standard cases $1 \leq n_3 \leq 3$ use the SAT step. The extreme cases $n_3 = 0$ and $n_3 = 4$ are simpler and are resolved mostly by deterministic checks after the local enumeration. The local-signature definitions, orbit representatives, residual CNFs, and final audits are given in full in [3].

Table 5
Numerical profile of the standard K_{33} cases $1 \leq n_3 \leq 3$.

case	orbit	local sig.	global dist.	type-0 surv.	SAT br.	UNSAT br.	audit surv.	hard residual	final surv.
grid_1n3	–	227	441	100	75	25	0	0	0
grid_2n3	total	273	1444	102	80	22	0	0	0
grid_3n3	total	168	543	24	19	5	3	0	0
prism_1n3	total	466	882	200	150	48	1	2	0
prism_2n3	total	285	1444	102	80	22	3	0	0
prism_3n3	total	170	335	25	20	5	2	0	0

Theorem 2. *There is no exact decomposition of K_{33} into 57 blocks of orders 3, 4, and 5.*

Table 6
Enumeration outcomes for the extreme K_{33} cases $n_3 = 0$ and $n_3 = 4$.

$n_3 = 0$ pattern	local sig.	orbit reps.	global dist.	type-0 surv.	block-compat. surv.	final surv.
grid	369	14	79	38	35	0
prism	377	54	79	38	35	0

$n_3 = 4$ pattern	orbit	type-0 subcase	templates	branches	repeated-edge rej.	no type-1 cand.
prism	unique	disjoint	45	1494	1458	36
prism	unique	share-1	110	1424	1422	2
grid	$\mathcal{O}_1$	disjoint	15	480	470	10
grid	$\mathcal{O}_1$	share-1	80	1040	1040	0
grid	$\mathcal{O}_2$	disjoint	45	1494	1458	36
grid	$\mathcal{O}_2$	share-1	108	1404	1404	0

Proof sketch. The edge-count equations leave only the block vectors $(2, 7, 48)$ and $(6, 0, 51)$. The first vector is impossible by [1, Lemma 7.2]. For the second vector, the structural reduction above shows that every decomposition belongs to one of the ten cases determined by the grid/prism triangle pattern and $n_3 \in \{0, 1, 2, 3, 4\}$. Tables 5 and 6 record the exact finite exclusion of these cases. Every branch is rejected either by a structural edge-disjointness audit, by the absence of a type-1 completion candidate, or by a SAT UNSAT certificate for the reduced residual instance. Hence both possible 57-block vectors are impossible. □

Together with the general lower bound [1, Theorem 8.1], Theorem 2 reduces the proof of the next result to excluding exact decompositions with 58 blocks. The remaining 58-block case can be excluded by a direct counting argument.

Theorem 3.

$$C^{\xi}(33, \{3, 4, 5\}, 2) \geq 59.$$

Proof. A general lower bound in [1, Theorem 8.1], applied to $v = 33$, shows that fewer than 57 blocks are impossible. Theorem 2 excludes 57 blocks, so it remains to exclude exact decompositions with 58 blocks. If such a decomposition had block vector (α, β, γ), then

$$\alpha + \beta + \gamma = 58, \qquad 3\alpha + 6\beta + 10\gamma = 528.$$

The only nonnegative solutions are $(0, 13, 45)$ and $(4, 6, 48)$.

First consider $(0, 13, 45)$. At a vertex x, let β_x and γ_x be the numbers of quadruples and quintuples through x. The local degree equation is

$$3\beta_x + 4\gamma_x = 32,$$

so $\beta_x \equiv 0 \pmod 4$. If some vertex had $\beta_x = 8$, then the 24 vertices sharing a quadruple with x would each have $\beta_y \geq 4$, forcing at least $(8 + 24 \cdot 4)/4 = 26$ quadruples, a contradiction. Hence $\beta_x \in \{0, 4\}$ for every x. Since $\sum_x \beta_x = 4 \cdot 13$, exactly 13 vertices have $\beta_x = 4$; denote their set by H. Every quadruple is contained in H, and the 13 quadruples cover $13 \cdot 6 = 78 = \binom{13}{2}$ distinct edges. They therefore partition the edges of $K_H \cong K_{13}$. No quintuple can contain two vertices from H. For every $x \in H$, the local equation gives $\gamma_x = 5$, so the quintuples have at least $13 \cdot 5 = 65$ distinct incidences with H. Since each quintuple has at most one such incidence, this would require at least 65 quintuples, contradicting $\gamma = 45$.

Now consider $(4, 6, 48)$. At a vertex x, write $\alpha_x, \beta_x, \gamma_x$ for the local counts. Then

$$2\alpha_x + 3\beta_x + 4\gamma_x = 32,$$

so β_x is even. If $\beta_x \geq 4$, then the $3\beta_x$ vertices sharing a quadruple with x are distinct and each has $\beta_y \geq 2$, whence $\sum_z \beta_z \geq \beta_x + 2(3\beta_x) > 24 = 4\beta$, impossible. Thus $\beta_x \in \{0, 2\}$. The only vertices with nonzero (α_x, β_x) are of types $(1, 2), (2, 0), (3, 2), (4, 0)$. If their numbers are $\lambda_1, \lambda_2, \lambda_3, \lambda_4$, then

$$\lambda_1 + 2\lambda_2 + 3\lambda_3 + 4\lambda_4 = 3\alpha = 12, \qquad 2\lambda_1 + 2\lambda_3 = 4\beta = 24.$$

It follows that $\lambda_1 = 12$ and $\lambda_2 = \lambda_3 = \lambda_4 = 0$. Let U be the 12 vertices of type $(1, 2)$. All triangles and quadruples lie inside U, so they cover $4 \cdot 3 + 6 \cdot 6$ edges of K_U, leaving

$$\binom{12}{2} - 4 \cdot 3 - 6 \cdot 6 = 18$$

edges inside U to be covered by quintuples. Let n_s be the number of quintuples meeting U in s vertices. Then

$$\sum_s n_s = 48, \qquad \sum_s s n_s = 12 \cdot 6 = 72, \qquad \sum_s \binom{s}{2} n_s = 18.$$

But $\binom{s}{2} \geq s - 1$ for $0 \leq s \leq 5$, so

$$18 = \sum_s \binom{s}{2} n_s \geq \sum_s (s-1) n_s = 72 - 48 = 24,$$

a contradiction. Hence neither 58-block vector is possible, and the lower bound follows. □

6. Conclusion

The results in this paper are meant to support one point: in clique-covering problems, construction, lower-bound exclusion and verification should often be separated. DLX and ILP are both useful for finding decompositions, but their relative performance changes with the instance. This is visible already in the K_{18} and K_{19} constructions and is consistent with the scalability data. SAT is used for a different purpose: after arithmetic, symmetry and local reductions, it proves that the remaining lower-bound branches are infeasible.

The role of the K_{33} computation is to show that the same organisation still works in a case where a direct model is not enough. The transferable part is not the particular grid/prism split, but the way the proof is assembled: reduce the possible parameters, enumerate branches up to symmetry, pass the remaining tasks to the solver best suited to them, and check every decoded output at the level of edge coverage. This is why the paper presents the K_{18}, K_{19} and K_{33} computations together.

Acknowledgments

We thank the reviewers for their careful reading and helpful remarks. This work is partially supported by VSB–Technical University of Ostrava via SGS grants No. SP2026/019 and No. SP2026/050. Petr Kovář and Yifan Zhang are co-funded by the European Union under the REFRESH–Research Excellence For Region Sustainability and High-tech Industries project, No. CZ.10.03.01/00/22 003/0000048, through the Operational Programme Just Transition.

Declaration on Generative AI

The authors have not employed any Generative AI tools.

Data and code availability

The computational artifacts supporting this paper are available in the archived datasets cited in the text: the scalability archive [10], the K_{18} DLX construction archive [11], the K_{19} ILP construction archive [12], the reduced SAT archive for K_{18} and K_{19} [13], the triangle benchmark [9], the K_{33} decomposition-exclusion archive [14], and the K_{33} monolithic benchmark archive [15].

References

[1] P. Kovář and Y. Zhang. Minimum-excess coverings of complete graphs by 3-, 4-, and 5-cliques. Preprint, arXiv:2603.18803v2, 2026. doi:10.48550/arXiv.2603.18803.

[2] P. Kovář and Y. Zhang. Optimal decomposition of K_{18} and K_{19} into K_3 and K_4. Preprint, arXiv:2507.06745, 2025. doi:10.48550/arXiv.2507.06745.

[3] P. Kovář and Y. Zhang. A SAT-based filtering framework for exact coverings of K_{33} by cliques of order 3, 4 or 5. Preprint, arXiv:2603.29548, 2026. doi:10.48550/arXiv.2603.29548.

[4] C. J. Colbourn and J. H. Dinitz, editors. *Handbook of Combinatorial Designs*. Chapman & Hall/CRC, second edition, 2006.

[5] D. E. Knuth. Dancing links. In *Millennial Perspectives in Computer Science*, pages 187–214. Palgrave, 2000.

[6] A. Biere, M. Heule, H. van Maaren, and T. Walsh, editors. *Handbook of Satisfiability*. IOS Press, 2009.

[7] A. Biere, K. Fazekas, M. Fleury, and M. Heisinger. CaDiCaL, Kissat, Paracooba, Plingeling, and Treengeling entering the SAT Competition 2020. In *Proceedings of SAT Competition 2020: Solver and Benchmark Descriptions*, pages 50–53, 2020.

[8] P. Kovář and Y. Zhang. Covering complete graphs using the dancing links algorithm. In *Proceedings of the 23rd Conference Information Technologies – Applications and Theory*, CEUR Workshop Proceedings 3498, pages 161–167, 2023.

[9] Y. Zhang. Benchmark data for decomposing complete graphs into K_3. Zenodo, 2026. doi:10.5281/zenodo.19139244.

[10] Y. Zhang. Scalability data for hybrid exact searches for $\{K_3, K_4\}$-coverings of complete graphs. Zenodo, 2026. doi:10.5281/zenodo.20176934.

[11] Y. Zhang. Reproducibility archive for the DLX construction of an optimal exact $\{K_3, K_4\}$-covering of K_{18}. Zenodo, 2026. doi:10.5281/zenodo.19361680.

[12] Y. Zhang. Reproducibility archive for the ILP construction of an optimal exact $\{K_3, K_4\}$-covering of K_{19}. Zenodo, 2026. doi:10.5281/zenodo.19361693.

[13] Y. Zhang. Reproducibility archive for reduced SAT computations on exact $\{K_3, K_4\}$-coverings of K_{18} and K_{19}. Zenodo, 2026. doi:10.5281/zenodo.19369963.

[14] Y. Zhang. Computational archive for the exclusion of a decomposition of K_{33} into 51 K_5 and 6 K_3. Zenodo, 2026. doi:10.5281/zenodo.19066645.

[15] Y. Zhang. Benchmark data and scripts for the K_{33} monolithic exact-cover experiments. Zenodo, 2026. doi:10.5281/zenodo.19076269.