A Strategy for Lowering the Upper Bound of R(5,5)

Thibault Gauthier

Czech Technical University in Prague, Czech Republic

Abstract

We present an approach for lowering the upper bound of R(5,5) by decomposing the problem into an extremely large number of configurations, each solvable by a SAT solver. To reduce the total number of cases, we apply generalization techniques that allow many configurations to be handled in a single SAT instance.

Introduction The Ramsey number $R(b, r)$ is the smallest integer n such that every red-blue coloring of the edges of the complete graph K_n on n vertices contains either a blue-clique of size b or a red-clique of size r. A colored graph (of size k) that does not contain any blue-clique of size b and any red-clique of size r is called an $R(b, r)$-graph ($R(b, r, k)$-graph) and is said to respect the $R(b, r)$-constraint. It has been known since 1989 that an $R(5, 5, 42)$-graph exists and giving the lower bound $R(5, 5) \geq 43$ [4]. $R(4, 5) = 25$ [6], which was proved in 1995, implies that $R(5, 5) \leq 50$. This was improved to $R(5, 5) \leq 48$ [1] in 2017 and further to $R(5, 5) \leq 46$ [2] in 2024.

In this work, we outline a strategy aimed at lowering the upper bound further. The approach assumes the existence of an $R(5, 5, 43)$-graph, chooses a vertex in K_{43} of blue-degree d (referred to as a splitting vertex) and decomposes the graph into its blue neighbors (an $R(4, 5, d)$-graph) and its red neighbors (an $R(5, 4, 42 - d)$-graph). A direct approach is to exhaustively generate all possible $R(4, 5, d)$-subgraphs and all $R(5, 4, 42 - d)$-graphs. A *gluing* problem consists of finding a color assignment for the transverse edges and respects the $R(5, 5)$-constraint. If all gluing problems constructed from all pairs of subgraphs are unsatisfiable, then $R(5, 5) \leq 43$. Further details on constructions and experimental results are provided in Appendix A.

Construction of a Gluing Problem There are an estimated 2.91×10^{19} [6] $R(4, 5)$-graphs and a complete list of such graphs has yet to be compiled. While our approach does not require full enumeration, it is helpful to describe the problem as if we were generating all $R(4, 5)$-graphs. A natural strategy is to decompose each candidate graph into the blue neighbors of a vertex, forming an $R(3, 5)$-graph, and the red neighbors, forming an $R(4, 4)$-graph. Since both $R(3, 5)$-graphs and $R(4, 4)$-graphs have been fully enumerated, all $R(4, 5)$-graphs (and, by symmetry, all $R(5, 4)$-graphs) can, in principle, be constructed by gluing such pairs while ensuring the $R(4, 5)$-constraint is satisfied.

To illustrate this, we construct a specific gluing problem P_0, consisting of a vertex of blue-degree 20, an $R(4, 5, 20)$-graph G_0 and an $R(5, 4, 22)$-graph H_0. G_0 is built by picking a splitting vertex of blue-degree 9, a random $R(3, 5, 9)$-graph for its blue neighbors and a random $R(4, 4, 10)$-graph for its red neighbors. Similarly, H_0 is built from a splitting vertex of blue-degree 10, a random $R(4, 4, 10)$-graph for its blue neighbors and a random $R(5, 3, 11)$-graph for its red neighbors. A possible assignment for the transverse edges in G_0 and H_0 is obtained by calling the SAT solver CaDiCaL [3].

Number of Gluing Problems The degree d of the primary splitting vertex can be restricted based on the following observations. In any red-blue coloring of a complete graph with an odd number of vertices, at least one vertex must have an even blue-degree. Therefore, we can assume that d is even without loss of generality. Due to the symmetry between red and blue, we may assume without loss of generality that the blue-degree is less than the red-degree. The blue-degree d and red-degree $42 - d$ must both be strictly less than 25, since $R(4,5) = R(5,4) = 25$. Combining these constraints, we are left with two cases: $d = 18$ and $d = 20$. Based on the estimated number of $R(4,5)$-graphs given in [6], the number of gluing problems is 1.73×10^{24} for $d = 18$ and 1.62×10^{34} for $d = 20$.

Proof of a Gluing Problem The SAT solver CaDiCaL is not able to prove that P_0 is unsatisfiable within a day. We therefore decompose P_0 into 622746 subproblems by considering separately each of the way the secondary splitting vertex in G_0 can be connected to vertices in H_0 while respecting the $R(5,5)$-constraint. Over the course of 200 CPU-days, CaDiCaL successfully proved that all subproblems are unsatisfiable, thus proving that P_0 is itself unsatisfiable. Assuming this instance is representative, applying the same exhaustive method to all 1.62×10^{34} gluing problems would require approximately $200 \times 1.62 \times 10^{34}$ CPU-days, which is an entirely impractical amount.

Generalization of a Gluing Subproblem The number of gluing problems is too large to be solved individually. Therefore, we introduce a generalization procedure that enables solving many subproblems simultaneously. The idea is to start with a specific subproblem Q_0 (a partially colored graph) and progressively forget the color of the edges in Q_0 to form more abstract versions Q_1,Q_2,...,Q_n. Each abstracted problem may cover subproblems from multiple gluing problems, allowing proofs to be reused. To decide which edge to generalize (i.e., make uncolored), we compute a score: the ratio of solving time (measured with `CaDiCaL`) to the number of subproblems covered (estimated by `ganak` [7] or a custom probabilistic model counter for problems that are too hard for `ganak`). We select the edge with the lowest score, which maximizes the number of subproblems solved per second. This greedy process continues until proving unsatisfiability becomes too slow (i.e., exceeds a time threshold).

In one experiment, starting from an easy subproblem Q_0 of P_0, solvable in 0.14 seconds, we were able to abstract 298 out of 403 edges before the runtime exceeded 20 seconds. This solves an estimated 2.4×10^{27} non-isomorphic subproblems. This estimate was obtained by dividing the total number of subproblems covered by Q_{298} (approximately 4.4×10^{36}) by the average number of subproblems per isomorphic class (approximately 1.8×10^{9}).

Covering Procedure From the remaining colored edges of the strongest abstraction Q_n, we create a blocking clause preventing us from revisiting any covered subproblems in future generalizations. The generalization procedure is then restarted from a different subproblem not covered by our previous blocking clauses. This subproblem may potentially come from a different gluing problem. This process repeats until all subproblems across all are covered. Completing this process would constitute a proof that $R(5,5) \leq 43$.

If we optimistically assume that our generalization experiment is representative, then the number of generalizations needed would be $(1.62 \times 10^{34}) \times 622746 \div (2.4 \times 10^{27}) = 4.2 \times 10^{12}$ times to complete the proof. Given the current cost of 10 CPU-hours per generalization, the proof could be completed in 42 trillion CPU-hours.

Verification We aim to verify the correctness of our algorithms using an interactive theorem prover, following the same methodology that led to the formal proof of $R(4,5) = 25$ [5].

Resources The code for our project is available at `https://github.com/barakeel/ramsey`.

References

[1] Vigleik Angeltveit and Brendan D. McKay. R(5, 5) ≤ 48. *J. Graph Theory*, 89(1):5–13, 2018.

[2] Vigleik Angeltveit and Brendan D McKay. R(5, 5) ≤ 46. *arXiv preprint arXiv:2409.15709*, 2024.

[3] Armin Biere, Tobias Faller, Katalin Fazekas, Mathias Fleury, Nils Froleyks, and Florian Pollitt. CaDiCaL 2.0. In Arie Gurfinkel and Vijay Ganesh, editors, *Computer Aided Verification - 36th International Conference, CAV 2024, Montreal, QC, Canada, July 24-27, 2024, Proceedings, Part I*, volume 14681 of *Lecture Notes in Computer Science*, pages 133–152. Springer, 2024.

[4] Geoffrey Exoo. A lower bound for r(5, 5). *J. Graph Theory*, 13(1):97–98, 1989.

[5] Thibault Gauthier and Chad E. Brown. A formal proof of r(4, 5)=25. In Yves Bertot, Temur Kutsia, and Michael Norrish, editors, *15th International Conference on Interactive Theorem Proving, ITP 2024, September 9-14, 2024, Tbilisi, Georgia*, volume 309 of *LIPIcs*, pages 16:1–16:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2024.

[6] Brendan D. McKay and Stanislaw P. Radziszowski. $R(4, 5) = 25$. *J. Graph Theory*, 19(3):309–322, 1995.

[7] Shubham Sharma, Subhajit Roy, Mate Soos, and Kuldeep S. Meel. Ganak: A scalable probabilistic exact model counter. In *Proceedings of International Joint Conference on Artificial Intelligence (IJCAI)*, 8 2019.

A Appendix

A.1 Initial Split

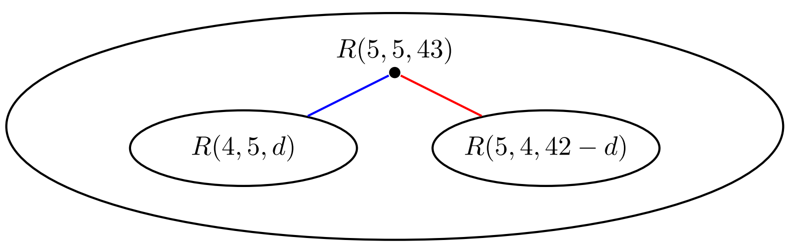

Figure 1: Decomposition of an $R(5,5,43)$-graph into a splitting vertex, its blue neighbors and its red neighbors.

A.2 Statistics for the Proof of P_0

Time (seconds)	0-0.25	0.25-0.5	0.5-1.0	1-2	2-4	4-8	8-16	16-32	32-64	64-128	128-
# Subproblems	137357	144628	59406	82570	53228	40028	33985	16678	18781	12946	23139

Table 1: Number of subproblems of P_0 solved in a given amount of time. The longest time taken by `CaDiCaL` to solve a subproblem is 8 hours.

A.3 Adjacency Matrices for the Different Stages

The adjacency matrices below represent partial graphs (problems). The unassigned edges are left blank, the blue edges are represented by `o`, the red edges are represented by `-`.

```
# o o o o o o o o o o o o o o o o o o o o - - - - - - - - - - - - - - - - - - - - - -
o # o o o o o o o o o - - - - - - - - - -
o o # - - o - - - - -
o o - # - - - - o - o
o o - - # - - o - o -
o o o - - # o - - - -
o o - - - o # - - o o
o o - - o - - # o - o
o o - o - - - o # o -
o o - - o - o - o # -
o o - o - - o o - - #
o -                   # o - - - - - - - o
o -                   o # - - - - - o o -
o -                   - - # - o - o o - o
o -                   - - - # o o o - o -
o -                   - - o o # - o - o -
o -                   - - - o - # o o o o
o -                   - - o o o o # - - o
o -                   - o o - - o - # o o
o -                   - o - o o o - o # -
o -                   o - o - - o o o - #
-                                         # o o o o o o o o o o - - - - - - - - - - -
-                                         o # o o o - o o - - o
-                                         o o # - o o o o o - -
-                                         o o - # o o o - o o -
-                                         o o o o # o - - o - -
-                                         o - o o o # - o - o -
-                                         o o o o - - # - - o -
-                                         o o o - - o - # - - o
-                                         o - o o o - - - # - o
-                                         o - - o - o o - - # -
-                                         o o - - - - - o o - #
-                                         -                     # o o o o - - o o o o
-                                         -                     o # o o - o o o o o -
-                                         -                     o o # - o o o o o - o
-                                         -                     o o - # o o o o - o o
-                                         -                     o - o o # o o - o o o
-                                         -                     - o o o o # o - o o -
-                                         -                     - o o o o o # o - - o
-                                         -                     o o o o - - o # - - o
-                                         -                     o o o - o o - - # o -
-                                         -                     o o - o o o - - o # -
-                                         -                     o - o o o - o o - - #
```

Figure 2: The splitting vertex (0) and beginning of the construction of the graph G_0 and H_0 by gluing blue neighbors and red neighbors of secondary splitting vertices

```
# o o o o o o o o o o o o o o o o o o o o - - - - - - - - - - - - - - - - - - - - - -
o # o o o o o o o o o - - - - - - - - - -
o o # - - o - - - - - o - o o o o - - - o
o o - # - - - - o - o - o o o o - - o - -
o o - - # - - o - o - - o - - - o o o - -
o o o - - # o - - - - o o o o - - - - o -
o o - - - o # - - o o o - o - - o o - o -
o o - - o - - # o - o - o - - - o - - o o
o o - o - - - o # o - - - o o - - - - o o
o o - - o - o - o # - o o - - o - o - o o
o o - o - - o o - - # o - - - o - o o o o
o - o - - o o - - o o # o - - - - - - - o
o - - o o o - o - o - o # - - - - - o o -
o - o o - o o - o - - - - # - o - o o - o
o - o o - o - - o - - - - - # o o o - o -
o - o o - - - - - o o - - o o # - o - o -
o - o - o - o o - - - - - - o - # o o o o
o - - - o - o - - o o - - o o o o # - - o
o - - o o - - - - - o - o o - - o - # o o
o - - - - o o o o o o - o - o o o - o # -
o - o - - - - o o o o o - o - - o o o - #
-                                         # o o o o o o o o o o - - - - - - - - - - -
-                                         o # o o o - o o - - o - o - - o o - - o - o
-                                         o o # - o o o o o - - - o o o - o - o - - -
-                                         o o - # o o o - o o - - - o - - - o - - o o
-                                         o o o o # o - - o - - - o o - o - - - - o o
-                                         o - o o o # - o - o - - o - o o - o - o o -
-                                         o o o o - - # - - o - o - o o - o - - o - o
-                                         o o o - - o - # - - o o o - - - - o o o - -
-                                         o - o o o - - - # - o o - o - o o - o - o -
-                                         o - - o - o o - - # - - - - o - o o - o o o
-                                         o o - - - - - o o - # o - - - o - o o o - o
-                                         - - - - - - o o o - o # o o o o - - o o o o
-                                         - o o - o o - o - - - o # o o - o o o o o -
-                                         - - o o o - o - o - - o o # - o o o o o - o
-                                         - - o - - o o - - o - o o - # o o o o - o o
-                                         - o - - o o - - o - o o - o o # o o - o o o
-                                         - o o - - - o - o o - - o o o o # o - o o -
-                                         - - - o - o - o - o o - o o o o o # o - - o
-                                         - - o - - - - o o - o o o o o - - o # - - o
-                                         - o - - - o o o - o o o o o - o o - - # o -
-                                         - - - o o o - - o o - o o - o o o - - o # -
-                                         - o - o o - o - - o o o - o o o - o o - - #
```

Figure 3: The three splitting vertices (0,1,21) and completed construction of G_0 and H_0. Proving that this partial matrix cannot be completed is the problem P_0.

```
# o o o o o o o o o o o o o o o o o o o o - - - - - - - - - - - - - - - - - - - - - -
o # o o o o o o o o o - - - - - - - - - - - - - - - - - - - - - - o o - - - o - - o - -
o o # - - o - - - - - o - o o o o - - - o
o o - # - - - - o - o - o o o o - - o - -
o o - - # - - o - o - - o - - - o o o - -
o o o - - # o - - - - o o o o - - - - o -
o o - - - o # - - o o o - o - - o o - o -
o o - - o - - # o - o - o - - - o - - o o
o o - o - - - o # o - - - o o - - - - o o
o o - - o - o - o # - o o - - o - o - o o
o o - o - - o o - - # o - - - o - o o o o
o - o - - o o - - o o # o - - - - - - - o
o - - o o o - o - o - o # - - - - - o o -
o - o o - o o - o - - - - # - o - o o - o
o - o o - o - - o - - - - - # o o o - o -
o - o o - - - - - o o - - o o # - o - o -
o - o - o - o o - - - - - - o - # o o o o
o - - - o - o - - o o - - o o o o # - - o
o - - o o - - - - - o - o o - - o - # o o
o - - - - o o o o o o - o - o o o - o # -
o - o - - - - o o o o o - o - - o o o - #
- -                                       # o o o o o o o o o o - - - - - - - - - - -
- -                                       o # o o o - o o - - o - o - - o o - - o - o
- -                                       o o # - o o o o o - - - o o o - o - o - - -
- -                                       o o - # o o o - o o - - - o - - - o - - o o
- -                                       o o o o # o - - o - - - o o - o - - - - o o
- -                                       o - o o o # - o - o - - o - o o - o - o o -
- -                                       o o o o - - # - - o - o - o o - o - - o - o
- -                                       o o o - - o - # - - o o o - - - - o o o - -
- -                                       o - o o o - - - # - o o - o - o o - o - o -
- -                                       o - - o - o o - - # - - - - o - o o - o o o
- -                                       o o - - - - - o o - # o - - - o - o o o - o
- o                                       - - - - - - o o o - o # o o o o - - o o o o
- o                                       - o o - o o - o - - - o # o o - o o o o o -
- -                                       - - o o o - o - o - - o o # - o o o o o - o
- -                                       - - o - - o o - - o - o o - # o o o o - o o
- -                                       - o - - o o - - o - o o - o o # o o - o o o
- o                                       - o o - - - o - o o - - o o o o # o - o o -
- -                                       - - - o - o - o - o o - o o o o o # o - - o
- -                                       - - o - - - - o o - o o o o o - - o # - - o
- o                                       - o - - - o o o - o o o o o - o o - - # o -
- -                                       - - - o o o - - o o - o o - o o o - - o # -
- -                                       - o - o o - o - - o o o - o o o - o o - - #
```

Figure 4: Adjacency matrix of the selected subproblem Q_0. It is an instantiation of P_0. There, the edges connecting the secondary splitting vertex 1 to H_0 have been assigned.

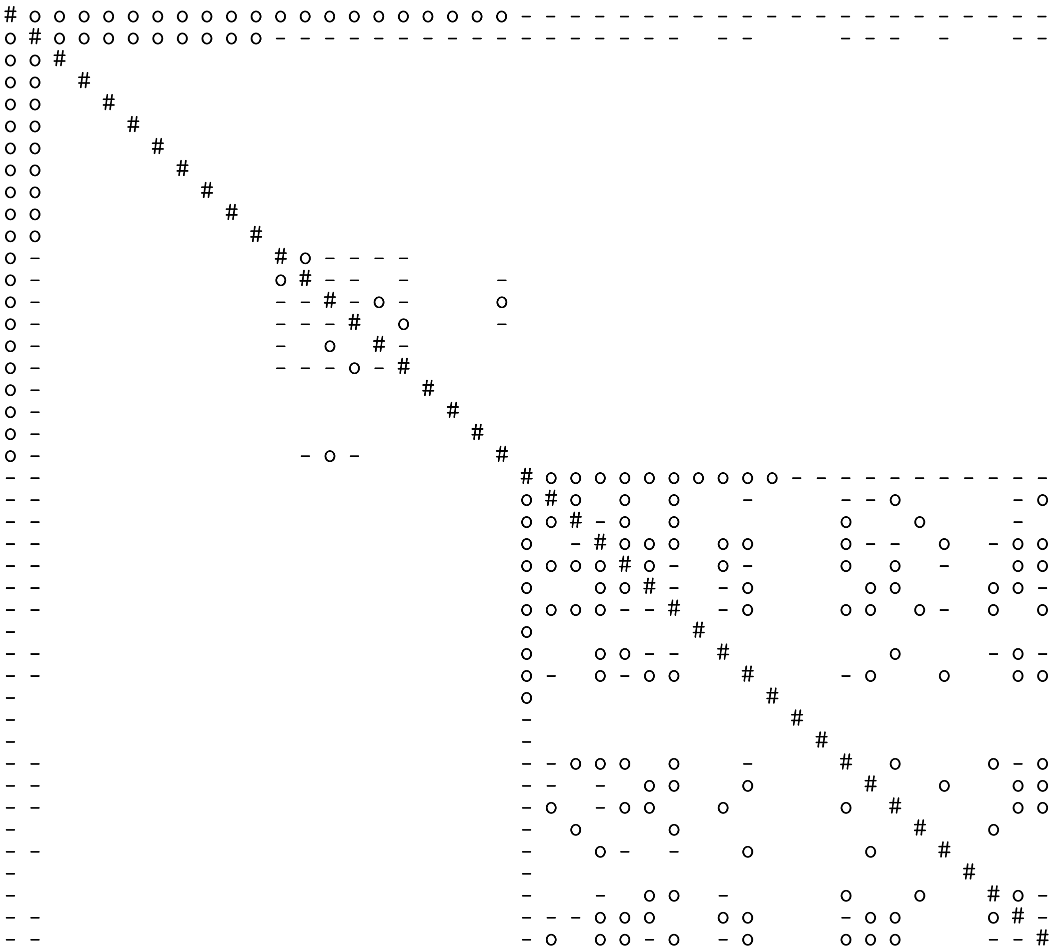

Figure 5: Adjacency matrix Q_{289} obtained by applying 289 abstraction steps to Q_0