Small Ramsey Numbers for Books, Wheels, and Generalizations

Bernard Lidický[a] Gwen McKinley[b] Florian Pfender[c]
Steven Van Overberghe[d]

Submitted: Nov 13, 2024; Accepted: Oct 21, 2025; Published: Dec 12, 2025

Abstract

We describe applications of a range of different computational methods to Ramsey numbers. We use flag algebras, local search, bottom-up generation, and enumeration of polycirculant graphs.

These methods are applied to Ramsey numbers for *books* and *wheels*. We also initiate the study of generalized small Ramsey numbers. Let $GR(r, K_s, t)$ denote the minimum number of vertices n such that any r-edge-coloring of K_n has a copy of K_s with at most t colors.

We establish over 20 new bounds including exact determination of the Ramsey numbers $R(W_5, W_7) = 15$, $R(W_5, W_9) = 18$, $R(B_2, B_8) = 21$, $R(B_3, B_7) = 20$, $GR(3, K_4, 2) = 10$, and $GR(4, K_4, 3) = 10$.

Mathematics Subject Classifications: 05D10, 05C55, 68R07

1 Introduction

The most classical problem in Ramsey theory deals with cliques in 2 colors: namely, it asks us to find the *Ramsey number* $R(K_s, K_t)$, the smallest number n guaranteeing that any red-blue edge-coloring of K_n must contain a red copy of K_s or a blue copy of K_t. Equivalently, one may ask for the largest n such that there exists a red-blue edge-coloring of K_{n-1} avoiding the "forbidden" subgraphs K_s in red and K_r in blue. There are many generalizations of this problem, including to larger numbers of colors, and forbidden subgraphs other than cliques. A dynamic survey by Radziszowski [31] tracks the state of

[a]Department of Mathematics, Iowa State University, U.S.A (`lidicky@iastate.edu`). Research supported by NSF DSM-2152490 and Scott Hanna Professorship.

[b]Center for Communications Research, La Jolla, U.S.A. (`gmckinle@ccr-lajolla.org`).

[c]Department of Mathematical and Statistical Sciences, University of Colorado Denver, U.S.A. (`Florian.Pfender@ucdenver.edu`). Research is partially supported by NSF grant DMS-2152498.

[d] Department of Mathematics, Computer Science and Statistics, Ghent University, Belgium (`steven.vanoverberghe@ugent.be`).

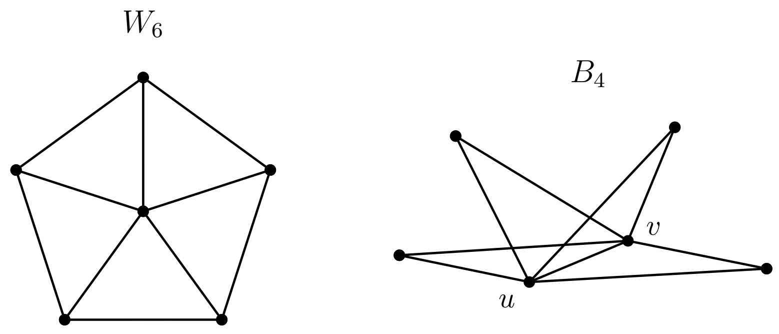

Figure 1: Wheel W_6 and book B_4.

knowledge in this area.

Here, we will focus on Ramsey numbers where the forbidden subgraphs are *wheels* or *books*. The **wheel** W_k consists of a cycle on $k-1$ vertices, together with a central vertex adjacent to all vertices of the cycle. The **book** B_k consists of $k+2$ vertices, an edge uv (the "spine"), and k vertices each adjacent only to u and v. See Figure 1 for an illustration.

There are a number of results on Ramsey numbers for books and wheels, summarized in the dynamic survey [31]. In particular, the first and third authors proved upper bounds for a variety of small Ramsey numbers, including for books and wheels, using the *flag algebra method* [25]; we establish most of the upper bounds here using the same method. The fourth author established a variety of lower bounds for graphs including wheels by explicit enumeration of *circulant* and *polycirculant* (or "block-circulant") graphs [19]; here, we use the same technique to establish a number of new lower bounds for books. In other recent papers, Ghebleh, Al-Yakoob, Kanso, and Stevanović [16] have used reinforcement learning and Wesley [38] has used SAT solvers to give Ramsey lower bound constructions for books and wheels.

We also explore a generalization of Ramsey numbers introduced by Erdős and Shelah [9, 10]. In classical Ramsey problems, one is looking for a monochromatic copy of a graph H in an r-edge-coloring of a graph K_n. The generalization asks for a copy of H with at most t colors for a fixed t. We investigate the case where H is a clique. Let $GR(r, K_s, t)$ denote the minimum number of vertices n such that any r-edge-coloring of K_n has a copy of K_s with at most t colors. Observe

$$GR(r, K_s, 1) = R(\underbrace{K_s, \ldots, K_s}_{r}).$$

These generalized Ramsey numbers have been studied asymptotically; for some of the literature, see [4, 20, 11, 30, 8]. For small values of r, s, t, $GR(r, K_s, t)$ has previously remained unexplored.

In the next section, we summarize our results. In Section 3 we describe our methods in more detail (bottom-up generation, flag algebras, polycirculant graphs, and tabu search). We finish with open questions.

2 Results

We prove the bounds presented in Table 1 below; asterisks indicate bounds that are tight, and the methods used to produce each bound are marked by letters as described in the caption of Table 1.

Note: we have recently become aware of independent work by Yanbo [40] establishing the same (tight) lower bound on $R(W_5, W_9)$ using an explicit construction. In [39], some of the results below on Ramsey numbers for books are also proven, and there is also a generalization of the second case of Theorem 1 to an infinite family. Specifically, a tight lower bound on $R(B_{n-1}, B_n)$ is given when $2n-1$ is a prime power congruent to 1 modulo 4.

	Lower (new)	Upper (new)	Lower (old)	Upper (old)
$R(W_5, W_7)$		15* (b/f)	15 [38]	16 [25]
$R(W_5, W_9)$	18* (b/p/t)	18* (b)	17 [31]	
$R(B_2, B_8)$	21* (b/p/t)	21* (b)	19 [34]	22 [14]
$R(B_2, B_9)$	22 (p/t)		21 [34]	24 [14]
$R(B_2, B_{10})$	25 (p)		23 [34]	26 [14]
$R(B_3, B_7)$	20⋆ (p/t)	20⋆ (f)		
$R(B_4, B_7)$	22 (p/t)	23 (f)		
$R(B_5, B_6)$	23* (p/t)			23 [34]
$R(B_5, B_7)$	25* (p)			25 [34]
$R(B_6, B_7)$	27* (p/t)			27 [34]
$R(B_6, B_8)$	29* (p)			29 [34]
$R(B_7, B_8)$	31* (p)			31 [34]
$R(B_8, B_8)$	33* (p)			33 [34]
$GR(3, K_4, 2)$	10* (b/t)	10* (b/f)		
$GR(3, K_5, 2)$	20 (t)	23 (f)		
$GR(3, K_6, 2)$	32 (t)	54 (f)		
$GR(4, K_4, 2)$	15 (t)	17 (f)		
$GR(4, K_4, 3)$	10* (b/t)	10* (b/f)		

Table 1: New upper and lower bounds on Ramsey numbers. Letters indicate the method used to produce each bound: (b) bottom-up generation, (f) flag algebras, (p) polycirculant graph generation, and (t) tabu search. When more than one letter is given, the bound was produced independently by the corresponding methods.

We also prove the following result covering the diagonal and almost-diagonal cases for some additional sizes of books. Note that some of these bounds are also included above in Table 1 for readers' convenience.

Theorem 1. For all $4 \leqslant n \leqslant 21$ we have

- $4n - 3 \leqslant R(B_{n-2}, B_n)$
- $R(B_{n-1}, B_n) = 4n - 1$
- $4n + 1 \leqslant R(B_n, B_n) \leqslant 4n + 2$

Note that it was also already known that $R(B_{n-2}, B_n) \leqslant 4n - 3$ if $n \equiv 2 \pmod 3$, and that $R(B_n, B_n) \leqslant 4n + 1$ if $4n + 1$ is not the sum of two squares.

2.1 Overview of methods and further details

Each of the lower bounds in Table 1 above is established by a construction provided in Appendix A. The constructions were found using tabu search or as polycirculant graphs, and verified independently in SageMath [35]; the code for books and wheels is available at https://github.com/gwen-mckinley/ramsey-books-wheels and https://github.com/Steven-VO/circulant-Ramsey. The additional lower bounds given in Theorem 1 were also established by polycirculant graph constructions, which are available in the same repository. The majority of the upper bounds were found by semidefinite programming using the flag algebra method, as described in [25]. The certificates for the upper bounds are available at http://lidicky.name/pub/gr. And several upper bounds are from "bottom-up generation," as described below.

Here is more detail on the structure of some lower bound constructions for generalized Ramsey numbers: for $GR(3, K_4, 2)$, one extremal coloring is composed of the Paley graph on 9 vertices in the first color, and the other two colors are 3 triangles each, see Figure 2(a). This is the only coloring with more than 6 symmetries. For $GR(4, K_4, 3)$, the only maximal graph is isomorphic to $3K_3$ in every color, see Figure 2(b). Joining two arbitrary colors, we find the graph described for $GR(3, K_4, 2)$. One example for $GR(3, K_5, 2)$ is the coloring composed of the (isomorphic) circulant graphs $C(19, [1, 7, 8])$, $C(19, [2, 3, 5])$ and $C(19, [4, 6, 9])$, see Figure 2(c).

3 Methods

3.1 Bottom-up generation

For small Ramsey numbers, it is possible to generate all Ramsey graphs by starting from a trivial case (for example the graph on one vertex) and repeatedly adding an extra vertex in every possible way without creating a subgraph isomorphic to one of the avoided graphs. For two-color Ramsey numbers this can be achieved efficiently with the program *geng* from the *nauty* package written by Brendan McKay [28]. We wrote a custom plugin

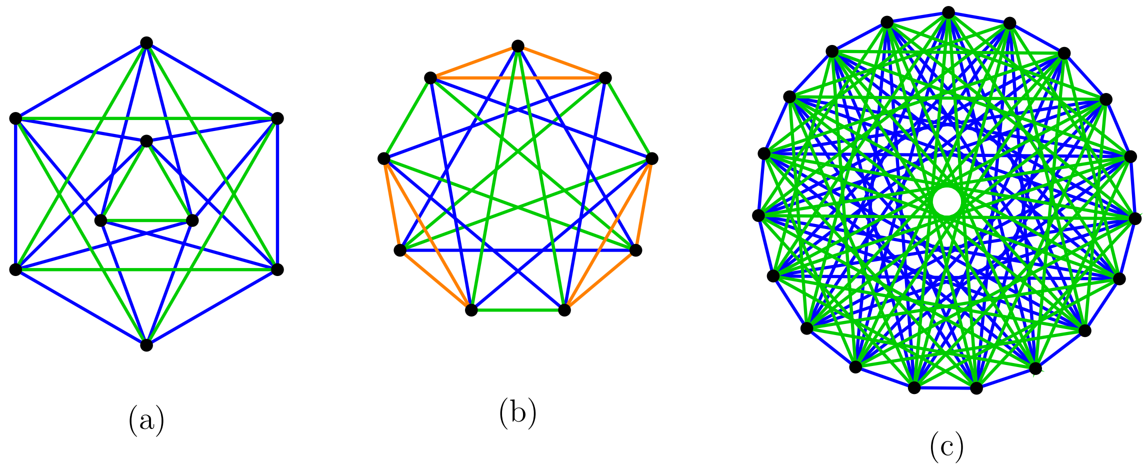
Figure 2: (a) A coloring of K_9 showing $GR(3, K_4, 2) \geqslant 10$. (b) The coloring of K_9 showing $GR(4, K_4, 3) \geqslant 10$. (c) A coloring of K_{19} showing $GR(3, K_5, 2) \geqslant 20$. Edges of one color omitted in all drawings for readability.

for generating Ramsey graphs avoiding wheels and books. This allowed us to find the values of $R(W_5, W_7)$ and $R(W_5, W_9)$ in 90 CPU-seconds and 36 CPU-hours respectively. The computation of $R(B_2, B_8)$ was significantly heavier and required approximately 2 years of CPU time. The counts of Ramsey graphs produced by bottom-up generation are presented in Table 2. To test the latter program, we also generated all $R(B_2, B_6)$-graphs (which requires only a trivial modification to the program) and verified that the results were identical to those in [6] by Black, Leven, and Radziszowski.

For the generalized Ramsey numbers, multiple colors are needed, which is not directly supported by *nauty*. We used a simple implementation in SageMath [35] to do the generation; the code is available at: `https://github.com/gwen-mckinley/ramsey-books-wheels`. Both $GR(3, K_4, 2)$ and $GR(4, K_4, 3)$ could be solved in less than one minute in this way.

3.2 Flag Algebras

The new upper bounds (except for $R(B_2, B_8)$ and $R(W_5, W_9)$) were obtained (independently) using the flag algebra method. This method was developed by Razborov [32] in a very general setting of model theory. To a graph theorist, a flag algebra is an algebra of subgraph densities in an unknown limit object of a sequence of graphs related to a question. In this algebra, one sets up a semidefinite program to find new bounds, certified by a typically large semidefinite matrix. This certificate is then translated into a sum of squares to save space and effort. Part of the input is the size of the small subgraphs one considers. Larger subgraphs usually allow for better bounds, but the program quickly reaches the limits of computational power due to combinatorial explosion, even for large computers.

The flag algebra method is naturally applicable to problems where one is interested in asymptotic results as it operates on the limit object. For example, the Turán density of small hypergraphs and its variants [5] is a group of problems where flag algebras led to many new improvements.

The Ramsey problems in this paper are not asymptotic, but instead deal with fixed-sized graphs; so the flag algebra method does not apply directly. Instead, we consider a balanced blow-up of the Ramsey graph to transfer the problem from a small finite setting into a limit object, using an additional edge color inside the blow-up sets. All forbidden subgraphs in the Ramsey problem are still forbidden in the blow-up, and the order of the Ramsey graph corresponds to the edge density of the additional color. This approach was developed by the first and third author in [25], we refer to that paper for more details. Here, we utilize the machinery developed there to obtain additional upper bounds. To solve the semidefinite programs, we used CSDP [7] and MOSEK [3]. The solvers produce numerical results, and we created tools to transform these results into exact arithmetic solutions in SageMath [35].

The main limitation of the flag algebra method is obtaining the solutions to the semidefinite programs. Typically, the semidefinite programs coming from applications of the flag algebra method can be solved when the number of unlabeled graphs, after application of color symmetry, is up to few hundred thousand when using a supercomputer. For generalized Ramsey numbers, we were able to compute on up to 7 vertices. For books and wheels, we were able to use 9 vertices. The semidefinite programs and lists of configurations used to construct them are available at https://lidicky.name/pub/gr. Most calculations needed less than one week of time. The largest one was for $GR(3, K_6, 2)$. It has 389,794 configurations on 7 vertices and took several months to solve using CSDP.

3.3 Polycirculant graphs

k-polycirculant graphs are graphs of order n that have a non-trivial automorphism α such that all vertex orbits under α have the same size n/k. For many Ramsey numbers, there exist extremal graphs that are polycirculant. One might also believe that if an extremal graph is unique, it is more likely to have some symmetries. For example, the unique Ramsey-(B_2, B_8)-graph on 20 vertices has 240 automorphisms and is 4-polycirculant.

We used the exhaustive algorithm outlined by Goedgebeur and the fourth author in [19] and adapted it slightly to generate all polycirculant Ramsey graphs avoiding books. With this generator, it is for example easy to show that there are exactly seven 2-polycirculant Ramsey-(B_2, B_9)-graphs on 20 vertices and exactly one 3-polycirculant Ramsey-(B_2, B_{10})-graph on 24 vertices (up to isomorphism). For every generated graph, we checked whether it could be extended with an extra vertex such that the Ramsey property still holds, but this was never the case for the largest-found polycirculant graphs.

Proof of Theorem 1. The upper bounds were proven in [34]. The lower bounds are witnessed by 2-polycirculant graphs (as was in fact one of the examples in [34]), which are available at: https://github.com/gwen-mckinley/ramsey-books-wheels. The code used to generate these examples is available at: https://github.com/Steven-VO/circulant-Ramsey. □

Note: the polycirculant graph constructions we found (regarding Theorem 1) all have the extra property that the first circulant block (the subgraph induced by the first vertex-orbit of the 2-polycirculant automorphism) is isomorphic to the complement of the second circulant block. For most cases there were many 2-polycirculant Ramsey graphs with this property. For example: there are 581 non-isomorphic 2-polycirculant (B_{12}, B_{13})-graphs on 50 vertices. Most of these graphs had no extra automorphisms besides the ones implied by being 2-polycirculant. This makes it hard to deduce a pattern in these graphs that could generalize to an infinite family.

3.4 Tabu search

Tabu search is a widely-used metaheuristic algorithm for solving combinatorial optimization problems, introduced by Glover [17]; for a survey of this method, see [18] or [15]. In this section, we give a description of tabu search in the context of Ramsey numbers, and discuss the details of how we used it to prove lower bounds.

3.4.1 Tabu search details

To prove a Ramsey lower bound n, our goal is to construct an edge-coloring of K_{n-1} avoiding forbidden monochromatic subgraphs (or, in the case of the generalized Ramsey numbers, forbidden subgraphs with too few edge colors). For simplicity, we will restrict the discussion here to the traditional 2-color case, but the ideas are exactly the same for the generalized problem.

In this setting, tabu search works roughly as follows: we start with a random 2-edge-coloring of K_{n-1} (equivalently, a random graph on $n-1$ vertices), and count the number of monochromatic substructures we are trying to avoid – this is the "score" of the graph. Then at each step, we re-color one edge, choosing whichever would produce the lowest-scoring graph, while rejecting any choice that would produce a graph already visited. This restriction is enforced by use of a "tabu set," which may consist either of forbidden moves (edges recently edited) or of whole graphs/colorings already visited. Here we take the second approach; more details and discussion are given below. For general background on tabu search, see

There are many variants of tabu search used in practice, but the one implemented here is very simple, and has two main features:

1. We do not restrict the length of the tabu – i.e., we reject any moves that would produce a graph already visited at *any* point during the search.

2. We allow the search to run indefinitely. We run several instances of the search in parallel, starting from independently sampled random graphs.

This approach has two main advantages: first, because this algorithm has no hyperparameters to tune, it is simple and usable "out of the box" for new problems. Second, by allowing the search to run indefinitely, we avoid inadvertently restarting too quickly. Empirically, it seems that the number of steps required to find a successful construction grows quickly with the problem parameters, and it is not immediately obvious how to predict this value. If we "bid low" and restart the search too quickly, we may consistently cut it short before it is able to reach a promising area of the search space. On the other hand, we observed no clear empirical disadvantage in allowing the search to continue[1]. Indeed, under some conditions, local search algorithms provably achieve optimal speedup via parallelization (as opposed to restarts); this is dependent on the runtime distribution of the particular algorithm. See Section 4.4 and Chapter 6 of [21] for further discussion and related empirical results.

We now return to the details of the choice of tabu set. Most variants of tabu search used in the literature store only a list of recent moves (e.g., edges recently edited), rather than whole graphs/configurations, due in large part to memory constraints [18, 21]. However, if only moves are stored, it is important to limit the length of the tabu set – otherwise all possible moves will quickly be forbidden. On the other hand, storing a full representation of each graph visited (and checking the result of each possible move against this set) is very computationally expensive. To avoid this issue, we instead store the graphs by hashing – see below for further details. This offers a much lower computational cost, while still allowing us to avoid all graphs previously visited. (It should be noted, however, that hash collisions may cause us to avoid some other graphs as well – we did not observe issues with this in practice, but it may be worth further study.) This general strategy has been seen very occasionally in the literature (see [26] by Lu, Martínez-Gavara, Hao, and Lai); but again, it is more common to store a tabu set of recent moves, in which case the length of the tabu is an important hyperparameter, which must be tuned correctly or dynamically updated for good performance.

Here are some additional details about our hashing-based strategy: in implementing this approach, an important design decision is the specific choice of hash function. In this algorithm, we must very often re-compute the hash of a graph after editing the color of one edge; to make this operation as efficient as possible, we chose the following hash function on edge-colored graphs. For each edge (i, j) in the graph, we record its color c and compute a hash[2] of the tuple $((i, j), c)$. Then, to combine these into a hash for the whole graph, we compute the bitwise XOR of the hashes of the individual (colored) edges. Notice that the hash of the graph can be updated quickly: if we change the color of an edge (i, j) from c to c', we can update the graph's hash by simply computing the bitwise

[1]There are a number of open-source hyperparameter tuning tools supporting this kind of experimentation; here, we used Optuna [1].

[2]In our implementation, we simply used the default Python hash function for tuples.

XOR of the old hash with the "tuple hashes" of $((i,j),c)$ and $((i,j),c')$.

As a final note, even after hashing, memory will eventually become a limitation if the search runs for a very large number of steps. In practice, we did not find this to be a major obstacle, but it is worth bearing in mind; and for very long searches, some additional effort may be needed to control memory usage.

3.4.2 Scoring functions

An important ingredient in any local search algorithm is an efficient "scoring" function; here, this means counting the number of "forbidden" substructures in a graph. In fact, rather than re-scoring the entire graph at each step, it is generally more efficient to compute the *change* in score at each step, since most of the graph will be unaffected by editing a single edge.

For books B_k, both the "scoring" and "score change" operations are quite efficient, and the cost is basically independent of k for $k \geqslant 2$. Explicitly, choosing a copy of B_k in a graph G reduces to first choosing a "spine" $(u,v) \in E(G)$, then choosing any k vertices from the common neighborhood $N(u) \cap N(v)$; see Figure 3.

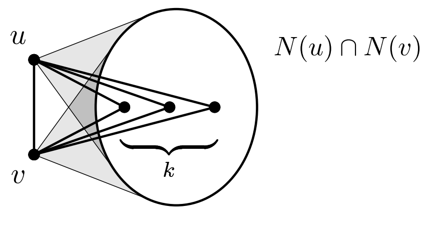

Figure 3: Book B_k in a graph.

Notice that for the purpose of counting, we do not need to list all such sets of k vertices; we just need to compute the binomial coefficient $\binom{|N(u)\cap N(v)|}{k}$. Given the value $|N(u) \cap N(v)|$, this operation is very efficient (certainly $O(|V(G)|)$), and in practice, all relevant values of the binomial coefficient can simply be stored in a lookup table. Using similar observations, we can also efficiently compute the "score change" for one step in the search (i.e., the change in the number of books from re-coloring an edge), though the details of the implementation are more involved.

For wheels W_k, we don't exploit any especially "nice tricks," and the cost of our scoring and score change functions are $\Omega(|V(G)|^k)$ and $\Omega(|V(G)|^{k-2})$ respectively – i.e., roughly the cost of enumerating over all subgraphs of the relevant size (where n is the number of vertices in the host graph G). Note, however, that significant improvement may be possible; counting wheels reduces largely to counting cycles, where surprisingly efficient algorithms are known (see [2] by Alon, Yuster, Raphael, and Zwick).

For generalized Ramsey numbers $GR(r, K_s, t)$, the situation is slightly different; here, we need to count copies of K_s with at most t colors (in an r-edge-colored complete graph G). Because clique counting is a well-studied problem in non-edge-colored graphs, we use the following strategy: (1) reduce to clique counting in a collection of non-edge-colored graphs, then (2) apply the "Pivoter" algorithm for fast clique-counting by Jain and Seshadhri [24] to each graph in the collection. This algorithm also facilitates counting cliques that contain a particular edge, which is useful for our "score change" computations. (Note: our code implementing this strategy is a rough prototype, so we have not made it publicly available, but it requires relatively minimal adaptations of the code for books and wheels. There is also an open-source implementation of the Pivoter clique-counting algorithm available online by Jain [23].)

In slightly more detail, we define the scoring function for the generalized Ramsey problem as follows: for each possible set $\mathcal{C}$ of t colors, we construct the (non-edge-colored) graph $G_{\mathcal{C}}$ consisting of edges whose colors in the original coloring G belong to $\mathcal{C}$. Then we take the "score" of G to be the sum of the number of cliques K_s in each of these graphs, that is, $\sum_{\mathcal{C}\in\binom{[r]}{t}}(\#\text{ copies of }K_s\text{ in }G_{\mathcal{C}})$. Notice that if a copy of K_s in G uses strictly *fewer* than t colors, it will be counted multiple times by this scoring function, once for each set $\mathcal{C}$ containing its colors. This could be avoided, for example by using inclusion-exclusion, but at a higher computational cost. Also, it is reasonable and perhaps even desirable that this scoring function more heavily penalizes copies of K_s using smaller numbers of colors – that is, copies that are further from satisfying the desired constraint.

4 Open questions

4.1 Upper bounds

In the flag algebras applications, we used the most straightforward version of the approach introduced in [25], which generalizes easily. But when calculating an upper bound for a particular Ramsey number, it is also possible to incorporate more constraints. In particular, one could try to include constraints on smaller graphs implied by smaller Ramsey numbers. For example when bounding $R(5,5)$, one may use $R(4,5) = 25$ when looking at neighborhoods of a fixed vertex. This approach has been explored by Volec [37], but it requires considerably more effort.

4.2 Lower bounds

4.2.1 Polycirculant graphs

It is interesting to ask whether Theorem 1 can be extended to larger values of n. We proved this lemma by starting a new search for polycirculant graphs for each n. There was no indication that the pattern would stop at $n \geqslant 22$; the generation simply started to take a lot of time. It seems plausible that there is some unified family of graphs giving this bound for all n. But again, as noted in Section 3.3, most of the graphs have no additional symmetry beyond being polycirculant, making it difficult to discern any larger pattern. Note that a partial answer to this question was given by Wesley [39] where the lemma is generalized to an infinite family consisting of orders that are twice a prime power (that is 1 modulo 4).

4.2.2 Tabu search

To extend the tabu search approach used here for finding lower bounds, there are a number of natural directions to explore. Perhaps most obvious would be to implement everything in a lower-level language – all the code for wheels and books is currently written in Python. In particular, the score-change functions dominate the total cost, and speeding them up would almost certainly improve performance; relevant here is an extensive literature on subgraph counting (see [33]), and it may be possible to simply incorporate existing open-source solutions.

Another small but potentially effective change could be to incorporate weights in the scoring function – that is, to penalize copies of one forbidden subgraph more than another, depending on their respective sizes and/or how recently they have been edited. This approach has often been used for Ramsey lower bounds, but there appears to be no clear theoretical justification for any particular choice of weights, and some papers have even arrived at contradictory conclusions [27, 13]. This question has been studied more extensively in the SAT literature (see [36, 22]), and it is possible some of this approach could be fruitfully adapted here.

Last but not least, one could try to be more strategic in choosing where to start a local search. One common approach is to build up from smaller graphs: we first find a construction on a smaller number of vertices avoiding "bad" subgraphs, then add a vertex, and run our local search starting from this new graph (as opposed to starting from a random graph), an approach successfully applied by McKay and Radziszowski [29] and Exoo [12]. The hope is that by doing this, we direct the search to a promising area of the (very large) search space. We also tried this method, with limited success, but it is likely worth further exploration.

Acknowledgments

The authors thank the anonymous referees for constructive remarks and suggestions on improving the presentation of this paper. The second author would like to thank David Hemminger and Larry Wilson for many helpful and stimulating discussions, and for their consistent encouragement. This work used the computing resources at the Center for Computational Mathematics, University of Colorado Denver, including the Alderaan cluster, supported by the National Science Foundation award OAC-2019089. We also used the Stevin computing infrastructure provided by the VSC (Flemish Supercomputer Center), funded by the Research Foundation Flanders (FWO) and the Flemish Government.

References

[1] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2019.

[2] Noga Alon, Raphael Yuster, and Uri Zwick. Finding and counting given length cycles. *Algorithmica*, 17(3):209–223, 1997. doi:10.1007/BF02523189.

[3] MOSEK ApS. *The MOSEK command line tools. Version 10.2.*, 2024. URL: https://docs.mosek.com/latest/cmdtools/index.html.

[4] Deepak Bal, Patrick Bennett, Emily Heath, and Shira Zerbib. Generalized Ramsey numbers of cycles, paths, and hypergraphs, 2024. arXiv:2405.15904.

[5] József Balogh, Felix Christian Clemen, and Bernard Lidický. Hypergraph Turán problems in ℓ_2-norm. In *Surveys in combinatorics 2022*, volume 481 of *London Math. Soc. Lecture Note Ser.*, pages 21–63. Cambridge Univ. Press, Cambridge, 2022. doi:10.1017/9781009093927.003.

[6] Kevin Black, Daniel Leven, and Stanisław P. Radziszowski. New bounds on some Ramsey numbers. *JCMCC. The Journal of Combinatorial Mathematics and Combinatorial Computing*, 78, 2011.

[7] Brian Borchers. CSDP, a C library for semidefinite programming. *Optim. Methods Softw.*, 11/12(1-4):613–623, 1999. Interior point methods. doi:10.1080/10556789908805765.

[8] Alex Cameron and Emily Heath. A $(5,5)$-colouring of K_n with few colours. *Combin. Probab. Comput.*, 27(6):892–912, 2018. doi:10.1017/S0963548318000172.

[9] Paul Erdős. Problems and results on finite and infinite graphs. In *Recent Advances in Graph Theory (Proc. Second Czechoslovak Sympos., Prague, 1974)*, pages 183–192. (loose errata). Academia, Prague, 1975.

[10] Paul Erdős. Solved and unsolved problems in combinatorics and combinatorial number theory. *Congr. Numer.*, 32:49–62, 1981.

[11] Paul Erdős and András Gyárfás. A variant of the classical Ramsey problem. *Combinatorica*, 17(4):459–467, 1997. doi:10.1007/BF01195000.

[12] Geoffrey Exoo. On the Ramsey number $r(4,6)$. *The Electronic Journal of Combinatorics*, 19(1):#P66, 2012. doi:10.37236/2102.

[13] Geoffrey Exoo. On some small classical Ramsey numbers. *The Electronic Journal of Combinatorics*, 20(1):#P68, 2013. doi:10.37236/3137.

[14] Ralph J. Faudree, Cecil C. Rousseau, and John Sheehan. Strongly regular graphs and finite Ramsey theory. *Linear Algebra and its Applications*, 46:221–241, 1982. doi:10.1016/0024-3795(82)90037-4.

[15] Michel Gendreau and Jean-Yves Potvin. *Tabu Search*, pages 41–59. Springer US, Boston, MA, 2010. doi:10.1007/978-1-4419-1665-5_2.

[16] Mohammad Ghebleh, Salem Al-Yakoob, Ali Kanso, and Dragan Stevanović. Reinforcement learning for graph theory, II. Small Ramsey numbers, 2024. arXiv:2403.20055.

[17] Fred Glover. Future paths for integer programming and links to artificial intelligence. *Computers & Operations Research*, 13(5):533–549, 1986.

[18] Fred Glover and Manuel Laguna. *Tabu Search*, pages 2093–2229. Springer US, Boston, MA, 1998. doi:10.1007/978-1-4613-0303-9_33.

[19] Jan Goedgebeur and Steven Van Overberghe. New bounds for Ramsey numbers $r(k_k - e, k_l - e)$. *Discrete Applied Mathematics*, 307:212–221, 2022. doi:10.1016/j.dam.2021.10.023.

[20] Enrique Gomez-Leos, Emily Heath, Alex Parker, Coy Schwieder, and Shira Zerbib. New bounds on the generalized Ramsey number $f(n, 5, 8)$. *Discrete Mathematics*, 347(7):114012, July 2024. doi:10.1016/j.disc.2024.114012.

[21] Holger H. Hoos and Thomas Stützle. *Stochastic Local Search*. The Morgan Kaufmann Series in Artificial Intelligence. Morgan Kaufmann, San Francisco, 2005. doi:10.1016/B978-1-55860-872-6.X5016-1.

[22] Abdelraouf Ishtaiwi, Feda Alshahwan, Naser Jamal, Wael Hadi, and Muhammad AbuArqoub. A dynamic clause specific initial weight assignment for solving satisfiability problems using local search. *Algorithms*, 14(1), 2021. doi:10.3390/a14010012.

[23] Shweta Jain. Pivoter. https://github.com/sjain12/Pivoter, 2020.

[24] Shweta Jain and C. Seshadhri. The power of pivoting for exact clique counting. In *Proceedings of the 13th International Conference on Web Search and Data Mining*, WSDM '20, page 268–276, New York, NY, USA, 2020. Association for Computing Machinery. doi:10.1145/3336191.3371839.

[25] Bernard Lidický and Florian Pfender. Semidefinite programming and Ramsey numbers. *SIAM Journal on Discrete Mathematics*, 35(4):2328–2344, 2021. doi:10.1137/18M1169473.

[26] Zhi Lu, Anna Martínez-Gavara, Jin-Kao Hao, and Xiangjing Lai. Solution-based tabu search for the capacitated dispersion problem. *Expert Systems with Applications*, 223, 2023. doi:10.1016/j.eswa.2023.119856f.

[27] Jeremy Mange and Andrew Dunn. Luus-Jaakola optimization procedure for Ramsey number lower bounds. *Int. J. Math. Comput. Sci.*, 10(1):57–68, 2015.

[28] Brendan D. McKay and Adolfo Piperno. Practical graph isomorphism, II. *Journal of Symbolic Computation*, 60:94–112, 2014. doi:10.1016/j.jsc.2013.09.003.

[29] Brendan D. McKay and Stanisław P. Radziszowski. Subgraph counting identities and Ramsey numbers. *Journal of Combinatorial Theory, Series B*, 69(2):193–209, 1997. doi:10.1006/jctb.1996.1741.

[30] Dhruv Mubayi. Edge-coloring cliques with three colors on all 4-cliques. *Combinatorica*, 18(2):293–296, 1998. doi:10.1007/PL00009822.

[31] Stanisław P. Radziszowski. Small Ramsey numbers. *Electron. J. Combin.*, 1, #DS1, 2024. doi:10.37236/21.

[32] Alexander Razborov. Flag algebras. *J. Symbolic Logic*, 72(4):1239–1282, 2007. doi:10.2178/jsl/1203350785.

[33] Pedro Ribeiro, Pedro Paredes, Miguel EP Silva, David Aparicio, and Fernando Silva. A survey on subgraph counting: Concepts, algorithms, and applications to network motifs and graphlets. *ACM Computing Surveys (CSUR)*, 54(2):1–36, 2021. doi:10.1145/3433652.

[34] Cecil C. Rousseau and John Sheehan. On Ramsey numbers for books. *Journal of Graph Theory*, 2(1):77–87, 1978. doi:10.1002/jgt.3190020110.

[35] The Sage Developers. *SageMath, the Sage Mathematics Software System (Version 10.3)*, 2024. URL: https://www.sagemath.org.

[36] John Thornton, Duc Nghia Pham, Stuart Bain, and Valnir Ferreira Jr. Additive versus multiplicative clause weighting for SAT. In *AAAI*, volume 4, pages 191–196, 2004.

[37] Jan Volec. Personal communication.

[38] William J. Wesley. Algebraic and Boolean methods for computation and certification of Ramsey-type numbers, 2023.

[39] William J. Wesley. Lower bounds for book Ramsey numbers, 2024. arXiv:2410.03625.

[40] Yanbo Zhang. Personal communication.

A Counts and constructions

n	$R(B_2, B_8)$	$R(W_5, W_7)$	$R(W_5, W_9)$	$GR(3, K_4, 2)$	$GR(4, K_4, 3)$
1	1	1	1	1	1
2	2	2	2	1	1
3	4	4	4	3	3
4	9	11	11	9	7
5	22	31	31	34	11
6	69	130	130	154	12
7	255	675	723	428	1
8	1301	4868	6456	556	1
9	9297	38059	87977	263	1
10	96618	251377	1627532	0	0
11	1405777	878658	27891376		
12	25330324	932411	250459368		
13	443322144	141871	509767930		
14	5130980404	1161	139131233		
15	?	0	46736023		
16	?		18956337		
17	?		272891		
18	?		0		
19	41				
20	1				
21	0				

Table 2: Number of Ramsey graphs by order. The multicolor numbers are given up to color-swapping isomorphism. Because the calculations for $R(B_2, B_8)$ had to be split up into multiple processes, we have no accurate counts for some orders.

To display the graph6 strings below, we recommend utilizing drawing by House of Graphs at https://houseofgraphs.org/draw_graph.

- $R(W_5, W_7) \geqslant 15$.
  graph6 format of a 14-vertex graph with no W_5 as a subgraph and no W_7 in the complement:

```
Mav?Hwu]`ySZpyyg?
```

- $R(W_5, W_9) \geqslant 18$.
  graph6 format of a 17-vertex graph with no W_5 as a subgraph and no W_9 in the complement:

  ```
  PIL{eMI^Jqp[gXkp_|zx0aww
  ```

- $R(B_2, B_8) \geqslant 21$.
  graph6 format of a 20-vertex graph with no B_2 as a subgraph and no B_8 in the complement:

  ```
  SXgcISrdSaJQBJs_jp@CWFOV?q}HWOPbc
  ```

- $R(B_2, B_9) \geqslant 22$.
  graph6 format of a 21-vertex graph with no B_2 as a subgraph and no B_9 in the complement:

  ```
  TXhJ?ScLQoHAO]EhcLe_G_nAEXuiBSnW?]?w
  ```

- $R(B_2, B_{10}) \geqslant 25$.
  graph6 format of a 24-vertex graph with no B_2 as a subgraph and no B_{10} in the complement:

  ```
  W?bFFbw^@{BwgDsAl?lg@U_cl@GlDGUacDih?lTKApSgDqh
  ```

- $R(B_3, B_7) \geqslant 20$.
  graph6 format of a 19-vertex graph with no B_3 as a subgraph and no B_7 in the complement:

  ```
  REf`OcBMI@ozZSyaMGil?ABm_o|jOO
  ```

- $R(B_4, B_7) \geqslant 22$.
  graph6 format of a 21-vertex graph with no B_4 as a subgraph and no B_7 in the complement:

  ```
  TqmoKUaoq\bAK|]oMgORPWJYMCxGye`EmTcY
  ```

- $R(B_5, B_6) \geqslant 23$.
  graph6 format of a 22-vertex graph with no B_5 as a subgraph and no B_6 in the complement:

  ```
  U_Ya{gHmOv}QfaSGkhXLXoN]krJqbE^?dvdCkHso
  ```

- $R(B_5, B_7) \geqslant 25$.
  graph6 format of a 24-vertex graph with no B_5 as a subgraph and no B_7 in the complement:

  ```
   W|teicY@kY[EQsKWEJqIIde]`^BZr?zwhGnwaCv`LUHF_UJ
  ```

- $R(B_6, B_7) \geqslant 27$.
  graph6 format of a 26-vertex graph with no B_6 as a subgraph and no B_7 in the complement:

  ```
  YMQcwl`gEBbKaZK{SbPhN~BNnaA\Q_YyVQ{A]uwEoemTUhkBdkk\T@Y_
  ```

- $R(B_6, B_8) \geqslant 29$.
  graph6 format of a 28-vertex graph with no B_6 as a subgraph and no B_8 in the complement:

  ```
  [CpdaqkTdUYkYkLUQEmOtXB]qEYqE\XBNUORYqIXjGhbUTRBUTR`jAhgYsiXBVdR
  ```

- $R(B_7, B_8) \geqslant 31$.
  graph6 format of a 30-vertex graph with no B_7 as a subgraph and no B_8 in the complement:

  ```
  ]UWsqWecqRCeceqRKcsDjoUn_lJ_lLoUd[Dxj_jMmAkl[FXl[BXUmDkdjbZGl[ZXAtplcDjrZG
  ```

- $R(B_8, B_8) \geqslant 33$.
  graph6 format of a 32-vertex graph with no B_8 as a subgraph and no B_8 in the complement:

  ```
  _Uzrpy]RpNQ]q]xN]Rrq]‘DnAJ^AJJ‘DewPXvAHJ[GsuwPhUwRhJ[JsavB|KUwNhPZa}cavD|GavD|GPZq}c
  ```

- $GR(3, K_4, 2) = 10$.
  Adjacency matrix of a 3-edge-colored 9-vertex graph with each K_4 containing more than 2 colors. There are 263 such colorings (up to color permutations). See Figure 2(a) for another example.

  ```
  [[0,2,3,3,3,1,1,1,2]
   [2,0,1,1,2,1,3,3,2]
   [3,1,0,2,3,1,3,2,1]
   [3,1,2,0,1,3,1,2,3]
   [3,2,3,1,0,2,2,3,1]
   [1,1,1,3,2,0,2,3,3]
   [1,3,3,1,2,2,0,1,3]
   [1,3,2,2,3,3,1,0,1]
   [2,2,1,3,1,3,3,1,0]]
  ```

- $GR(4, K_4, 2) \geqslant 15$
  Adjacency matrix of a 4-edge-colored 14-vertex graph with each K_4 containing more than 2 colors.

```
[[0,4,3,1,2,3,4,3,2,2,4,1,4,1]
 [4,0,2,4,1,1,1,3,4,3,2,2,3,1]
 [3,2,0,2,2,3,1,1,4,1,3,3,4,4]
 [1,4,2,0,4,2,2,3,1,3,4,3,1,2]
 [2,1,2,4,0,4,3,4,2,1,1,3,2,1]
 [3,1,3,2,4,0,1,4,1,2,2,4,1,3]
 [4,1,1,2,3,1,0,4,3,2,3,3,4,2]
 [3,3,1,3,4,4,4,0,2,2,1,2,2,1]
 [2,4,4,1,2,1,3,2,0,4,1,1,3,3]
 [2,3,1,3,1,2,2,2,4,0,3,1,1,4]
 [4,2,3,4,1,2,3,1,1,3,0,4,2,3]
 [1,2,3,3,3,4,3,2,1,1,4,0,4,1]
 [4,3,4,1,2,1,4,2,3,1,2,4,0,2]
 [1,1,4,2,1,3,2,1,3,4,3,1,2,0]]
```

- $GR(4, K_4, 3) = 10$
  Adjacency matrix of a 4-edge-colored 9-vertex graph with each K_4 containing more than 3 colors. There is only one such graph on 9 vertices. See Figure 2(b).

```
[[0,2,3,3,1,4,2,1,4]
 [2,0,1,4,3,3,2,4,1]
 [3,1,0,3,4,2,4,2,1]
 [3,4,3,0,2,1,1,4,2]
 [1,3,4,2,0,3,4,1,2]
 [4,3,2,1,3,0,1,2,4]
 [2,2,4,1,4,1,0,3,3]
 [1,4,2,4,1,2,3,0,3]
 [4,1,1,2,2,4,3,3,0]]
```

- $GR(3, K_5, 2) \geqslant 20$
  Adjacency matrix of a 3-edge-colored 19-vertex graph with each K_5 containing more than 2 colors.

```
[[0,1,2,2,3,2,3,1,1,3,3,1,1,3,2,3,2,2,1],
 [1,0,1,2,2,3,2,3,1,1,3,3,1,1,3,2,3,2,2],
 [2,1,0,1,2,2,3,2,3,1,1,3,3,1,1,3,2,3,2],
 [2,2,1,0,1,2,2,3,2,3,1,1,3,3,1,1,3,2,3],
 [3,2,2,1,0,1,2,2,3,2,3,1,1,3,3,1,1,3,2],
 [2,3,2,2,1,0,1,2,2,3,2,3,1,1,3,3,1,1,3],
 [3,2,3,2,2,1,0,1,2,2,3,2,3,1,1,3,3,1,1],
 [1,3,2,3,2,2,1,0,1,2,2,3,2,3,1,1,3,3,1],
 [1,1,3,2,3,2,2,1,0,1,2,2,3,2,3,1,1,3,3],
 [3,1,1,3,2,3,2,2,1,0,1,2,2,3,2,3,1,1,3],
 [3,3,1,1,3,2,3,2,2,1,0,1,2,2,3,2,3,1,1],
 [1,3,3,1,1,3,2,3,2,2,1,0,1,2,2,3,2,3,1],
 [1,1,3,3,1,1,3,2,3,2,2,1,0,1,2,2,3,2,3],
 [3,1,1,3,3,1,1,3,2,3,2,2,1,0,1,2,2,3,2],
 [2,3,1,1,3,3,1,1,3,2,3,2,2,1,0,1,2,2,3],
 [3,2,3,1,1,3,3,1,1,3,2,3,2,2,1,0,1,2,2],
 [2,3,2,3,1,1,3,3,1,1,3,2,3,2,2,1,0,1,2],
 [2,2,3,2,3,1,1,3,3,1,1,3,2,3,2,2,1,0,1],
 [1,2,2,3,2,3,1,1,3,3,1,1,3,2,3,2,2,1,0]]
```

- $GR(3, K_6, 2) \geqslant 32$
  Adjacency matrix of a 3-edge-colored 31-vertex graph with each K_6 containing more than 2 colors.

```
[[0,3,3,3,3,2,1,2,1,1,1,1,3,1,3,2,2,2,3,3,1,3,2,1,2,2,3,3,2,1,1]
 [3,0,2,1,3,2,3,2,2,1,2,2,2,1,1,1,2,2,2,2,1,3,3,3,1,1,1,1,3,3,2]
 [3,2,0,2,1,1,3,3,1,3,2,2,3,3,2,2,1,1,3,1,3,2,1,3,3,1,1,1,1,2,2]
 [3,1,2,0,1,3,1,3,3,1,1,2,3,1,2,3,1,1,1,3,1,2,2,3,2,2,3,3,3,2,1]
 [3,3,1,1,0,1,1,2,1,1,2,1,1,2,3,2,1,2,3,2,2,3,2,2,1,2,1,3,3,3,2]
 [2,2,1,3,1,0,1,2,2,3,1,3,2,2,3,3,1,3,2,2,1,3,2,3,1,1,1,3,1,1,3]
 [1,3,3,1,1,1,0,1,1,2,3,1,3,3,2,2,3,1,1,3,2,2,3,1,1,3,2,2,1,3,1]
 [2,2,3,3,2,2,1,0,2,3,2,3,3,3,1,1,2,2,1,2,3,1,1,3,1,1,3,2,3,2,1]
 [1,2,1,3,1,2,1,2,0,2,2,1,2,2,1,3,1,2,2,3,2,1,2,1,3,2,3,1,1,3,3]
 [1,1,3,1,1,3,2,3,2,0,2,2,3,1,3,1,1,1,1,1,2,1,3,3,3,2,2,2,3,2,1]
 [1,2,2,1,2,1,3,2,2,2,0,2,2,2,2,2,3,2,3,3,1,2,3,1,1,3,1,3,2,1,2]
 [1,2,2,2,1,3,1,3,1,2,2,0,1,3,3,3,1,3,1,2,3,1,1,1,1,2,1,2,3,2,3]
 [3,2,3,3,1,2,3,3,2,3,2,1,0,3,1,2,1,1,2,1,3,1,1,3,1,2,1,1,2,2,2]
 [1,1,3,1,2,2,3,3,2,1,2,3,3,0,1,1,1,3,2,2,3,2,2,2,2,1,1,1,3,1,3]
 [3,1,2,2,3,3,2,1,1,3,2,3,1,1,0,1,2,3,3,1,2,2,1,2,3,3,2,1,1,3,3]
 [2,1,2,3,2,3,2,1,3,1,2,3,2,1,1,0,3,3,3,3,1,1,3,3,1,1,3,2,2,3,2]
 [2,2,1,1,1,1,3,2,1,1,3,1,1,1,2,3,0,2,3,2,2,3,3,2,2,3,1,3,2,3,1]
 [2,2,1,1,2,3,1,2,2,1,2,3,1,3,3,3,2,0,1,1,3,3,2,2,2,1,1,3,1,1,3]
 [3,2,3,1,3,2,1,1,2,1,3,1,2,2,3,3,3,1,0,2,3,3,2,2,2,3,1,3,1,2,1]
 [3,2,1,3,2,2,3,2,3,1,3,2,1,2,1,3,2,1,2,0,2,1,1,2,3,1,3,1,1,2,2]
 [1,1,3,1,2,1,2,3,2,2,1,3,3,3,2,1,2,3,3,2,0,2,1,1,1,1,2,2,3,1,3]
 [3,3,2,2,3,3,2,1,1,1,2,1,1,2,2,1,3,3,3,1,2,0,1,3,1,1,2,3,3,3,1]
 [2,3,1,2,2,2,3,1,2,3,3,1,1,2,1,3,3,2,2,1,1,1,0,2,2,3,1,1,3,3,1]
 [1,3,3,3,2,3,1,3,1,3,1,1,3,2,2,3,2,2,2,2,1,3,2,0,2,2,2,3,3,1,2]
 [2,1,3,2,1,1,1,1,3,3,1,1,1,2,3,1,2,2,2,3,1,1,2,2,0,1,3,3,2,3,3]
 [2,1,1,2,2,1,3,1,2,2,3,2,2,1,3,1,3,1,3,1,1,1,3,2,1,0,1,3,2,2,2]
 [3,1,1,3,1,1,2,3,3,2,1,1,1,1,2,3,1,1,1,3,2,2,1,2,3,1,0,2,3,1,1]
 [3,1,1,3,3,3,2,2,1,2,3,2,1,1,1,2,3,3,3,1,2,3,1,3,3,3,2,0,1,2,1]
 [2,3,1,3,3,1,1,3,1,3,2,3,2,3,1,2,2,1,1,1,3,3,3,3,2,2,3,1,0,3,2]
 [1,3,2,2,3,1,3,2,3,2,1,2,2,1,3,3,3,1,2,2,1,3,3,1,3,2,1,2,3,0,3]
 [1,2,2,1,2,3,1,1,3,1,2,3,2,3,3,2,1,3,1,2,3,1,1,2,3,2,1,1,2,3,0]]
```