Some Problems in Graph Theory and Scheduling

Mingxian Zhong

Submitted in partial fulfillment of the
requirements for the degree
of Doctor of Philosophy
in the Graduate School of Arts and Sciences

COLUMBIA UNIVERSITY

2018

ABSTRACT

Some Problems in Graph Theory and Scheduling

Mingxian Zhong

In this dissertation, we present three results related to combinatorial algorithms in graph theory and scheduling, both of which are important subjects in the area of discrete mathematics and theoretical computer science. In graph theory, a graph is a set of vertices and edges, where each edge is a pair of vertices. A coloring of a graph is a function that assigns each vertex a color such that no two adjacent vertices share the same color. The first two results are related to coloring graphs belonging to specific classes. In scheduling problems, we are interested in how to efficiently schedule a set of jobs on machines. The last result is related to a scheduling problem in an environment where there is uncertainty on the number of machines.

The first result of this thesis is a polynomial time algorithm that determines if an input graph containing no induced seven-vertex path is 3-colorable. This affirmatively answers a question posed by Randerath, Schiermeyer and Tewes in 2002. Our algorithm also solves the list-coloring version of the 3-coloring problem, where every vertex is assigned a list of colors that is a subset of $\{1, 2, 3\}$, and gives an explicit coloring if one exists. This is joint work with Flavia Bonomo, Maria Chundnovsky, Peter Maceli, Oliver Schaudt, and Maya Stein.

A graph is H-free if it has no induced subgraph isomorphic to H. In the second part of this thesis, we characterize all graphs H for which there are only finitely many minimal non-three-colorable H-free graphs. This solves a problem posed by Golovach *et al.* We also characterize all graphs H for which there are only finitely many H-free minimal obstructions for list 3-colorability. This is joint work with Maria Chudnovsky, Jan Goedgebeur and Oliver Schaudt.

The last result of this thesis deals with a scheduling problem addressing the uncertainty regarding the machines. We study a scheduling environment in which jobs first need to be grouped into some sets before the number of machines is known, and then the sets need to be scheduled on machines without being separated. In order to evaluate algorithms in such an environment, we introduce the

idea of an α-robust algorithm, one which is guaranteed to return a schedule on any number m of machines that is within an α factor of the optimal schedule on m machines, where the optimum is not subject to the restriction that the sets cannot be separated. Under such environment, we give a $(\frac{5}{3}+\epsilon)$-robust algorithm for scheduling on parallel machines to minimize makespan, and show a lower bound of $\frac{4}{3}$. For the special case when the jobs are infinitesimal, we give a 1.233-robust algorithm with an asymptotic lower bound of 1.207. This is joint work with Clifford Stein.

Table of Contents

List of Figures

List of Tables

Acknowledgments

Firstly, I would like to express my deepest gratitude to my advisers, Maria Chudnovsky and Clifford Stein. Back in the days when I was in my master's program, Maria's class inspired my interest in graph theory, and the experience of working together with her was one of the most important reasons that I decided to apply for Ph.D. programs. During my Ph.D. program, both Maria and Cliff continually provided every bit of assistance I needed for my study and research. Under their guidance, I learned how to conduct a research and how to write the results as formal papers. They encouraged me to pursue an academic career after graduation and they are exactly the kind of professor I wanted to be. I am also very thankful for their support of my traveling to conferences around the world. It was truly a blessing to have both of them as my advisors.

Besides the advisors, I would also like to thank other coauthors I have been working with during my graduate study: Flavia Bonomo, Jan Goedgebeur, Shenwei Huang, Peter Maceli, Oliver Schaudt, Paul Seymour, Sophie Spirkl, Juraj Stacho and Maya Stein. I learned a lot from them and it's my great pleasure to write joint papers with them.

I would like to thank everyone in the IEOR department. All staff members here have always been helpful, patient and kind. It was my great opportunity to be able to learn from such knowledgeable faculty here. I am very thankful to Fei, Zhipeng, Ni, Yanan and Xinshang: It was a great memory of studying together with them in preparation for the qualifying exam, which is also probably the last qualifying exam in the department (or at least for a while).

Lastly and most importantly, I would like to thank my parents, Zhihua Zhong and Xiaoyan Wang, for their unconditional love, care and support all over the years. My mother had wished to attend my Ph.D. graduation ceremony, but unfortunately this wish can never become true as she passed away three years ago. I hereby dedicate this thesis to her.

In loving memory of my mother, Xiaoyan Wang (1963-2015).

Chapter 1

Introduction

In this dissertation, we present three results related to combinatorial algorithms in graph theory and scheduling, both of which are important subjects in the area of discrete mathematics and theoretical computer science. The first two results are related to coloring graphs belonging to specific classes and the last result is related to a scheduling problem addressing the environment where there is uncertainty on the number of machines. We will explain those results, and discuss some of the motivations and background in this section.

1.1 Coloring Graphs with Forbidden Induced Subgraphs

In graph theory, a graph G is a set of vertices and edges, where each edge is a pair of vertices. Graphs can be used to model many types of real-world systems, such as transportations, social networks and software design. However, many fundamental graph theory problems, such as the maximum independent set problem, the Hamiltonian path problem and the graph coloring problem, on a general graph are NP-complete, which means that in general it is hard to find efficient algorithms to solve those problems. So people are interested in solving those problems efficiently for graphs with some structural properties. The problem here we are specifically interested in is coloring graphs with forbidden induced subgraphs.

A *k-coloring* of a graph $G = (V, E)$ is a function $f : V \to \{1, \ldots, k\}$ such that $f(v) \neq f(w)$ whenever $vw \in E$. The *coloring problem*, whose input is a graph G and a natural number k, consists of deciding whether G is k-colorable or not. This well-known problem is one of Karp's 21 NP-

complete problems [36] (unless $k = 2$; then the problem is solvable in linear time). Stockmeyer [57] proved that the problem remains NP-complete even if $k \geq 3$ is fixed, and Maffray and Preissmann proved that it remains NP-complete for triangle-free graphs [41].

List variations of the vertex coloring problem can be found in the literature. For a survey on this kind of related problems, see [58]. In the *list-coloring problem*, every vertex v comes equipped with a list of permitted colors $L(v)$, and we require the coloring to respect these lists, i.e., $f(v) \in L(v)$ for every v in V. For a positive integer k, the *k-list-coloring problem* is a particular case in which $|L(v)| \leq k$ for each v in V, but the union of the lists can be an arbitrary set. If the size of the list assigned to each vertex is at most two (i.e., 2-list-coloring), the instance can be solved in $O(|V|+|E|)$ time [16; 19; 59], by reducing the problem to a 2-SAT instance, which Aspvall, Plass and Tarjan [5] showed can be solved in linear time (in the number of variables and clauses). The *list k-coloring problem* is a particular case of k-list-coloring, in which the lists associated to each vertex are a subset of $\{1, \ldots, k\}$. Since list k-coloring generalizes k-coloring, it is NP-complete as well.

Let H and G be graphs. We say that H is an *induced subgraph* of G if $V(H) \subseteq V(G)$, and $u, v \in V(H)$ are adjacent in H if and only if u, v are adjacent in G. we say that G *contains* H if some induced subgraph of G is isomorphic to H. If G does not contain H, we say that G is *H-free*. Because of the notorious hardness of k-coloring, efforts were made to understand the problem on restricted graph classes. Some of the most prominent such classes are the classes of H-free graphs. Kamiński and Lozin [35] and independently Král, Kratochvíl, Tuza, and Woeginger [37] proved that for any fixed $k, g \geq 3$, the k-coloring problem is NP-complete for the class of graphs containing no cycle of length less than g. As a consequence, if the graph H contains a cycle, then k-coloring is NP-complete for $k \geq 3$ for the class of H-free graphs.

The *claw* is the complete bipartite graph $K_{1,3}$. A theorem of Holyer [30] together with an extension due to Leven and Galil [39] imply that if a graph H contains a claw, then for every fixed $k \geq 3$, the k-coloring problem is NP-complete for the class of H-free graphs.

Combined, these two results only leave open the complexity of the k-coloring problem for the class of H-free graphs where H is a fixed acyclic claw-free graph, i.e., a disjoint union of paths. There is a nice recent survey by Hell and Huang on the complexity of coloring graphs without paths and cycles of certain lengths [27] and another nice survey by Golovach et al. [22]. We denote a path and a cycle on t vertices by P_t and C_t, respectively.

k\t	4	5	6	7	8	...
3	$O(m)$ [14]	$O(n^\alpha)$ [44]	$O(mn^\alpha)$ [49]	P [6]	?	...
4	$O(m)$ [14]	P [28]	P[12; 13]	NPC [31]	NPC	...
5	$O(m)$ [14]	P [28]	NPC [31]	NPC	NPC	...
6	$O(m)$ [14]	P [28]	NPC	NPC	NPC	...
⋮	⋮	⋮	⋮	⋮	⋮	⋱

Table 1.1: Table of known complexities of the k-coloring problem in P_t-free graphs.

The strongest known results related to our work are due to Huang [31], who proved that 4-coloring is NP-complete for P_7-free graphs, and that 5-coloring is NP-complete for P_6-free graphs. On the positive side, Hoàng, Kamiński, Lozin, Sawada, and Shu [28] have shown that k-coloring can be solved in polynomial time on P_5-free graphs for any fixed k. Recently, in a joint work with Chudnovsky and Spirkl [12; 13], we proved that 4-coloring is polynomial-time solvable for P_6-free graphs, which settles the last remaining open case of the complexity of k-coloring P_t-free graphs for any fixed $k \geq 4$. On the other hand, for $k = 3$ it is not known whether there exists a t such that 3-coloring is NP-complete for P_t-free graphs. Randerath and Schiermeyer [49] gave a polynomial time algorithm for 3-coloring P_6-free graphs. Later, Golovach et al. [23] showed that the list 3-coloring problem can be solved efficiently for P_6-free graphs. Some of these results are summarized in Table 1.1. Here '?' stands for open problems.

In Chapter 3 of this thesis, we show that the 3-coloring problem for P_7-free graphs is polynomial-time solvable, answering positively a question first posed in 2002 by Randerath et al. [49; 50]. This is joint work with Flavia Bonomo, Maria Chundnovsky, Peter Maceli, Oliver Schaudt, and Maya Stein, and first appeared in [6]. Our algorithm even works for the list 3-coloring problem. This is not trivial: there are cases where k-coloring and list k-coloring have different complexities (unless P = NP). For instance, in the class of P_6-free graphs, 4-coloring can be solved in polynomial time [12; 13] while list 4-coloring is NP-complete [32].

1.2 Obstruction to Coloring and List-Coloring

We say that a graph is *k-chromatic* if it is k-colorable but not $(k-1)$-colorable. A graph is called *k-critical* if it is k-chromatic, but every proper subgraph is $(k-1)$-colorable. The characterization of critical graphs is a notorious problem in the theory of graph coloring. Since it is NP-hard to decide whether a given graph admits a k-coloring for $k \geq 3$, there is little hope of giving a characterization of the $(k+1)$-critical graphs that is useful for algorithmic purposes. The picture changes if one restricts the structure of the graphs under consideration. Bruce *et al.* [7] proved that there are exactly six 4-critical P_5-free graphs. By using the characterization of 4-critical P_5-free graphs, Maffray and Morel [40] gave a linear time certifying algorithm of the problem of 3-coloring P_5-free graphs. On the other hand, Hoàng *et al.* [29] constructed an infinite family of k-critical P_5-free graphs for $k \geq 5$.

In a joint work with Chudnovsky, Goedgebeur and Schaudt [10], we proved that there are exactly 24 4-critical P_6-free graphs [10]. Moreover, together with counterexamples constructed, we were able to give a dichotomy theorem of the problem: for a connected H, there are finitely many 4-critical H-free graphs if and only if H is a subgraph of P_6. This solves a problem posed by Golovach *et al.* [24] and answers a question asked by Seymour [53]. Also, it implies the existence of a certifying algorithm of this problem.

In Chapter 4 of this thesis, we extend the above dichotomy theorem by removing the requirement that H is connected. Specifically, we proved that there are only finitely many H-free 4-vertex-critical graphs if and only if H is an induced subgraph of P_6, $2P_3$, or $P_4 + kP_1$ for some $k \in \mathbb{N}$. The tools we used in [10] were tailored specifically for the P_6-free case and do not generalize well, while our new approach is significantly more powerful. The idea is to transfer the problem to the more general list setting and solve it there. However, this generality comes at a certain cost: we can only give a very rough upper bound on the maximum number of vertices an H-free 4-vertex-critical graph can have, while in [10] we were able to give a picture of all critical graphs.

The second main result in this chapter is the analog of the above theorem in the list setting. To state it, we need the following concepts. Let L be a mapping that maps each vertex of G to a subset of $\{1, \ldots, k\}$. We say that the pair (G, L) is colorable if there is a k-coloring c of G with $c(v) \in L(v)$ for each $v \in V(G)$. As we are considering the obstructions to list 3-coloring, we call a pair (G, L) with $L(v) \subseteq \{1, 2, 3\}$, $v \in V(G)$, a minimal list-obstruction if (G, L) is not colorable

but for all induced proper subgraphs A of G the pair $\left(A, L|_{V(A)}\right)$ is colorable. We proved that there are only finitely many H-free minimal list-obstructions if and only if H is an induced subgraph of P_6 or $P_4 + kP_1$ for some $k \in \mathbb{N}$. This is joint work with Maria Chudnovsky, Jan Goedgebeur and Oliver Schaudt, and it first appeared in [11].

1.3 Scheduling with Uncertainty

For many problems, one does not know the entire input accurately and completely in advance. There are different ways of addressing such uncertainty, e.g. via online algorithms (assuming the input arrives over time), dynamic algorithms (assuming the input changes over time), stochastic optimization (assuming the input includes random variables) or robust optimization (assuming that there is bounded uncertainty in the data). Another way of addressing uncertainty is to require one solution that is good against all possible values of the uncertain parameters. Examples of work in this direction include the universal traveling salesman problem (one tour that is good no matter which subset of points arrive) [46], robust matchings (one matching is chosen and then evaluated by its top k edges, where k is unknown) [26; 42], a knapsack of unknown capacity(one policy of packing that is good irrespective of the actual capacity) [15] and 2-stage scheduling (some decisions must be made before the actual scenario is known) [9; 54]. In scheduling problems, there are many ways to model uncertainty in the jobs, including online algorithms [1; 2], in which the set of jobs is not known in advance, stochastic scheduling [45], in which the jobs are modeled as random variables, and work on schedules that are good against multiple objective functions [4; 43; 51]. But there is much less work studying the possibility of uncertainty in the machines, and the work we are aware of studies uncertainty in speed or reliability (breakdowns) [3; 17].

Motivated by the need to understand how to make scheduling decisions without knowing how many machines we will have, we consider a different notion of uncertainty – a scenario in which you don't know how many machines you are going to have, but you still have to commit (partially) to a schedule by making significant decisions about partitioning the jobs before knowing the number of machines.

This type of decision arises in a variety of settings. For example, many scheduling problems are fundamentally about packing items onto machines and there are many examples of problems

that concern packing items where there are multiple levels of commitment to be made with partial information. For example, in a warehouse, a large order may need to be placed into multiple boxes, without knowing exactly how many trucks there will be to ship the items. You therefore want to be able to pack the items well, given the various possible number of trucks. Another example involves problems in modern data centers. In data centers, there are some systems which require you to group work together into "bundles" without knowing exactly how many machines will be available. For example, in a map-reduce type computation, the mapping function naturally breaks the data into some number of groups g. However, there are some unknown numbers of available machines m, and you typically have to design your mapping function, choosing a g and associated grouping, without knowing m. You may know a range of possible values for m, or it may vary widely depending on the availability of machines at the time you run the map-reduce computation (and the availability is typically not under your control). As more and more computing moves to the "cloud", that is, moves to large shared data centers, we anticipate that this problem of grouping work without knowing the number of machines will become more widespread.

In Chapter 5 of this thesis, we consider one of the simplest scheduling problems – minimizing makespan on identical parallel machines. We choose this problem partly as a proof of concept for our two-stage model. We consider the following specific model. We are given a set of n jobs, J, with a known processing times $p(j)$ for each job j, and a number M, which is an upper bound on the number of machines we might have. An algorithm must commit, before knowing how many machines there are, to grouping the jobs into M *bags*, where each job is assigned to exactly one of the bags. We call this step the *packing* step. Only after completing the packing step do we learn the number of machines m. We now need to compute a schedule, with the restriction that we must keep the bags together, that is, we will assign one or more bags to each machine. We call this step the *scheduling* step. As in other robust problems, we want to do well against all possible numbers of machines. We therefore evaluate our schedule by the ratio of the makespan of our schedule, $ALG(m, M)$, to the makespan of a schedule that knew m in advance, opt_m, taking the worst case over all possible values of m. If an algorithm always provides a ratio of at most α, where $\alpha = \max_{1 \leq m \leq M} \frac{ALG(m,M)}{\text{opt}_m}$, we call it α-robust. (We may also consider scenarios in which there are different upper and lower bounds on the range of m; the definition of robustness extends in the obvious way.)

The main result of Chapter 5 is an algorithm for minimizing makespan on parallel machines. which is $(\frac{5}{3}+\epsilon)$-robust; and we show a lower bound of 4/3 on the robustness of any algorithm for minimizing makespan on parallel machines. This is joint work with Clifford Stein, and first appeared in the proceedings of *SODA* [56].

1.4 Outline

In this section, we give a brief outline of this thesis.

- In Chapter 2, we give some graph theory definitions that will appear later in this thesis.
- In Chapter 3, we give a polynomial time algorithm that determines if an input graph containing no induced seven-vertex path is 3-colorable. This is joint work with Flavia Bonomo, Maria Chundnovsky, Peter Maceli, Oliver Schaudt, and Maya Stein, and first appeared in *Combinatorica* [6].
- In Chapter 4, we characterize all graphs H for which there are only finitely many minimal non-three-colorable H-free graphs and all graphs H for which there are only finitely many H-free minimal obstructions for list 3-colorability. This is joint work with Maria Chudnovsky, Jan Goedgebeur and Oliver Schaudt, and it first appeared in [11].
- In Chapter 5, we give a $(\frac{5}{3}+\epsilon)$-robust algorithm for scheduling with uncertainty on the number of machines. This is joint work with Clifford Stein, and first appeared in the proceedings of *SODA* [56].

Chapter 2

Definitions

In this chapter, we present several definitions in graph theory which will appear throughout this thesis. All graphs in this thesis are finite and simple.

2.1 Basic Graph Definitions

For a graph G, we use $V(G)$ to denote its vertex set and $E(G)$ to denote its edge set. For $u, v \in V(G)$, we say that u and v are *adjacent* if $\{u, v\} \in E(G)$, and *non-adjacent* otherwise. A *clique* in a graph is a set of vertices such that every pair of the vertices are adjacent, and a *stable set* is a set of vertices all pairwise non-adjacent. The *neighborhood* of a vertex $v \in V(G)$ is the set of all vertices adjacent to v, and is denoted by $N_G(v)$, or simply $N(v)$ when there is no danger of confusion. Let X be a subset of $V(G)$. We denote by $G[X]$, or $G|X$ *the subgraph of* G *induced by* X, that is, the subgraph of G with vertex set X such that $u, v \in X$ are adjacent in $G[X]$ if and only if $\{u, v\} \in E(G)$. We denote by $G \setminus X$ the graph $G[V(G) \setminus X]$. If $X = \{v\}$ for some $v \in V(G)$, we write $G \setminus v$ instead of $G \setminus \{v\}$. Let H be a graph. If $G|X$ is isomorphic to H, then we say that X *is an* H *in* G. If G has no induced subgraph isomorphic to H, then we say that G is *H-free*. For a family $\mathcal{F}$ of graphs, we say that G is *$\mathcal{F}$-free* if G is F-free for every $F \in \mathcal{F}$. We write $G_1 + \ldots + G_k$ for the disjoint union of graphs $G_1, \ldots, G_k$. For a vertex set S, we use $N(S)$ to denote $(\bigcup_{v \in S} N(v)) \setminus S$.

For all t, let P_t denote the *path* on t vertices, which is the graph with vertex set $\{p_1, \ldots, p_t\}$ such that p_i is adjacent to p_j if and only if $|i - j| = 1$. Let C_t denote the *cycle* on t vertices, which is the graph with vertex set $\{p_1, \ldots, p_t\}$ such that p_i is adjacent to p_j if and only if $|i - j| = 1$ or

$t-1$. By convention, when explicitly describing a path or a cycle, we always list the vertices in order. Let G be a graph. When $G|\{p_1,\ldots,p_n\}$ is the path P_n, we say that p_1-...-p_n *is a* P_n *in* G. Similarly, when $G|\{c_1,c_2,\ldots,c_n\}$ is the cycle C_n, we say that c_1-c_2-...-c_n-c_1 *is a* C_n *in* G.

Let A and B be disjoint subsets of $V(G)$. For a vertex $b \in V(G)\setminus A$, we say that b *is complete to* A if b is adjacent to every vertex of A, and that b *is anticomplete to* A if b is non-adjacent to every vertex of A. If every vertex of A is complete to B, we say A *is complete to* B, and if every vertex of A is anticomplete to B, we say that A *is anticomplete to* B. If $b \in V(G)\setminus A$ is neither complete nor anticomplete to A, we say that b *is mixed on* A. The *complement* $\overline{G}$ of G is the graph with vertex set $V(G)$ such that two vertices are adjacent in $\overline{G}$ if and only if they are non-adjacent in G. If $\overline{G}$ is connected we say that G is *anticonnected*. For $X \subseteq V(G)$, we say that X is *connected* if $G|X$ is connected, and that X is anticonnected if $G|X$ is anticonnected. A *component* of $X \subseteq V(G)$ is a maximal connected subset of X, and an *anticomponent* of X is a maximal anticonnected subset of X. We write *component of* G to mean a component of $V(G)$. A subset D of $V(G)$ is called a *dominating set* if every vertex in $V(G)\setminus D$ is adjacent to at least one vertex in D; in this case we also say that $G|D$ is a *dominating subgraph* of G.

A *k-coloring* of a graph G is a mapping $c : V(G) \to \{1,\ldots,k\}$ such that if $x,y \in V(G)$ are adjacent, then $c(x) \neq c(y)$. If a k-coloring exists for a graph G, we say that G is *k-colorable*. We say that a graph is *k-chromatic* if it is k-colorable but not $(k-1)$-colorable. A graph is called *k-critical* if it is k-chromatic, but every proper subgraph is $(k-1)$-colorable. A graph is called *k-vertex-critical* if it is k-chromatic, but every proper induced subgraph is $(k-1)$-colorable. A graph is called *H-free, k-vertex-critical* if it is H-free, k-chromatic, but every proper H-free induced subgraph is $(k-1)$-colorable. A *list system* L of a graph G is a mapping which assigns each vertex $v \in V(G)$ a finite subset of $\mathbb{N}$, denoted by $L(v)$. A *subsystem* of a list system L of G is a list system L' of G such that $L'(v) \subseteq L(v)$ for all $v \in V(G)$. We say a list system L of the graph G has *order* k if $L(v) \subseteq \{1,\ldots,k\}$ for all $v \in V(G)$. In this article, we will only consider list systems of order 3. Notationally, we write (G,L) to represent a graph G and a list system L of G. We say that c, a coloring of G, is an *L-coloring* of G, or a *coloring of* (G,L) provided $c(v) \in L(v)$ for all $v \in V(G)$. We say that (G,L) is *colorable*, if there exists a coloring of (G,L). A *partial coloring* of (G,L) is a mapping $c : U \to \mathbb{N}$ such that $c(u) \in L(u)$ for all $u \in U$, where U is a subset of $V(G)$. Note that here we allow for edges uv of $G|U$ with $c(u) = c(v)$. If there is no such edge, we call c *proper*.

Let G be a graph and let L be a list system of order 3 for G. We say that (G, L) is a *list-obstruction* if (G, L) is not colorable. As stated earlier, we call (G, L) a *minimal list-obstruction* if, in addition, $(G \setminus x, L)$ is colorable for every vertex $x \in G$.

Let (G, L) be a list-obstruction. We say a vertex $v \in V(G)$ is *critical* if $G \setminus v$ is L-colorable and *non-critical* otherwise. If we repeatedly delete non-critical vertices of G to obtain a new graph, G' say, such that (G', L) is a minimal list obstruction, we say that (G', L) is a minimal list obstruction *induced* by (G, L).

Let u, v be two vertices of a list-obstruction (G, L). We say that u *dominates* v if $L(u) \subseteq L(v)$ and $N(v) \subseteq N(u)$. It is easy to see that if there are such vertices u and v in G, then (G, L) is not a minimal list-obstruction. We frequently use this observation without further reference.

2.2 Updating Lists

Let G be a graph and let L be a list system for G. Let $v, w \in V(G)$ be adjacent, and assume that $|L(w)| = 1$. To *update the list of v from w* means to delete from $L(v)$ the unique element of $L(w)$. If the size of the list of v is reduced to one, we sometimes say that v is *colored*, and refer to the unique element in the list of v as the *color* of v. Throughout the thesis, we make use of distinct updating procedures to reduce the sizes of the lists, and we define them below.

If $P = v_1\text{-}\ldots\text{-}v_k$ is a path and $|L(v_1)| = 1$, then to *update from v_1 along P* means to update v_2 from v_1 if possible, then to update v_3 from v_2 if possible, and so on. When v_k is updated from v_{k-1}, we stop the updating.

Let $X \subseteq V(G)$ such that $|L(x)| \leq 1$ for all $x \in X$. For a subset $A \subseteq V(G) \setminus X$, we say that we *update the lists of the vertices in A with respect to X* if we update each $a \in A$ from each $x \in X$. We say that we *update the lists with respect to X* if $A = V(G) \setminus X$. Let $X_0 = X$ and $L_0 = L$. For $i \geq 1$ define X_i and L_i as follows. L_i is the list system obtained from L_{i-1} by updating with respect to X_{i-1}. Moreover, $X_i = X_{i-1} \cup \{v \in V(G) \setminus X_{i-1} : |L_i(v)| \leq 1 \text{ and } |L_{i-1}(v)| > 1\}$. We say that L_i is obtained from L by *updating with respect to X i times*. If $X = \{w\}$ we say that L_i is obtained by *updating with respect to w i times*. If for some i, $W_i = W_{i-1}$ and $L_i = L_{i-1}$, we say that L_i was obtained from L by *updating exhaustively with respect to X (or w)*. For simplicity, if X is an induced subgraph of G, by updating with respect to X we mean updating with respect to $V(X)$.

We also adopt the following convention. If for some i, if two vertices of X_{i-1} with the same list are adjacent, or $L_i(v) = \emptyset$ for some v, we set $L_i(v) = \emptyset$ for every $v \in V(G) \setminus X_i$. Observe that in this case $(G|X_i, L_i)$ is not colorable, and so we have preserved at least one minimal list obstruction induced by (G, L).

Chapter 3

Three-coloring graphs without induced paths on seven vertices

3.1 Introduction and Preliminaries

Graph coloring is one of the fundamental problems in graph theory and also one of the most famous NP-complete problems. So we are interested in the complexity of coloring graph without certain induced subgraphs. It is known that k-coloring H-free graphs is NP-complete unless H is a disjoint union of paths [30; 35; 37; 39]. In this chapter, we prove the following main theorem, showing that 3-coloring and list 3-coloring problem for P_7-free graphs are polynomial-time solvable, which is one of the remaining cases of k-coloring P_t-free graphs (see Chapter 2 for formal definitions).

3.1.1. *One can decide whether a given P_7-free graph G has a list 3-coloring, and find such a coloring (if it exists) in polynomial time. The running time of the proposed algorithm is $O(|V(G)|^{21}(|V(G)|+|E(G)|))$.*

The algorithm given by Theorem 3.1.1 is based on the following ideas. First we apply some preprocessing techniques and compute a small 2-dominating set (i.e., a set such that every vertex has distance at most two to some vertex of the set). Then we use a controlled enumeration based on a structural analysis of the considered graphs, in order to reduce the problem to a polynomial number of instances of list 3-coloring in which the size of the list of each vertex is at most two. These instances, in turn, can be reduced to an instance of 2-SAT, which can be solved in linear

time in the number of variables and clauses [5], several authors [16; 19; 59] independently proved the following.

3.1.2. *If a list system L of a graph G is such that $|L(v)| \leq 2$ for all $v \in V(G)$, then a coloring of (G, L), or a determination that none exists, can be obtained in $O(|V(G)| + |E(G)|)$ time.*

Let G be a graph. A subset S of $V(G)$ is called *monochromatic* with respect to a given coloring c of G if $c(u) = c(v)$ for all $u, v \in S$. Let L be a list system of G, and Z a set of subsets of $V(G)$. We say that (G, L, Z) is *colorable* if there is a coloring c of (G, L) such that S is monochromatic with respect to c for all $S \in Z$.

A triple (G', L', Z') is a *restriction* of (G, L, Z) if

(a) G' is an induced subgraph of G,

(b) the list system L' is a subsystem of L restricted to the set $V(G')$, and

(c) Z' is a set of subsets of $V(G')$ such that if $S \in Z$ then $S \cap V(G') \subseteq S'$ for some $S' \in Z'$.

Let $\mathcal{R}$ be a set of restrictions of (G, L, Z). We say that $\mathcal{R}$ is *colorable* if at least one element of $\mathcal{R}$ is colorable. If $\mathcal{L}$ is a set of list systems of G, we write $(G, \mathcal{L}, Z)$ to mean the set of restrictions (G, L', Z) where $L' \in \mathcal{L}$.

Note that if two sets S and S' are monochromatic and have a non-empty intersection, then $S \cup S'$ is monochromatic, too. Thus, for each triple (G, L, Z) there is an equivalent triple (G, L, Z') such that Z' contains only mutually disjoint sets. During our algorithm, we compute the set family Z such that the sets are mutually disjoint. Under this assumption, the proof of Lemma 3.1.2 can be easily modified to obtain the following generalization [55].

3.1.3. *If a list system L of a graph G is such that $|L(v)| \leq 2$ for all $v \in V(G)$, and Z is a set of mutually disjoint subsets of $V(G)$, then a coloring of (G, L, Z), or a determination that none exists, can be obtained in $O(|V(G)| + |E(G)|)$ time.*

Proof. By traversing once each set in Z, create a vector r that maps each vertex v with a representative $r(v)$ on its set (the same representative for all the vertices in one set). Define $r(v) = v$ if v does not belong to a set in Z. Traversing the vector r once, iteratively for each $v \in V(G)$, update

$L(r(v)) = L(r(v)) \cap L(v)$. If at some point $L(r(v)) = \emptyset$, return that no coloring exists. These steps can be performed in $O(|V(G)|)$ time.

If none of the lists $L(r(v))$ is empty, compute the 2-SAT formula that expresses the coloring problem of (G, L, Z), similarly as for (G, L) in Lemma 3.1.2. Namely, define for each vertex $v \in V(G)$ and each color $j \in L(r(v))$ the variable $x_{r(v)j}$ to model that vertex v gets color j. Notice that if v and w are in the same set of Z, then $r(v) = r(w)$, thus the sets of Z will be monochromatic in every coloring derived from a solution of the formula.

If $L(r(v)) = \{j\}$, add $(x_{r(v)j})$ as a clause, and if $L(r(v)) = \{j, k\}$, add $(x_{r(v)j} \vee x_{r(v)k})$ as a clause. This ensures every vertex gets a color. Finally, for each edge $vw \in E(G)$ and each color $j \in L(r(v)) \cap L(r(w))$, add the clause $(\neg x_{r(v)j} \vee \neg x_{r(w)j})$. This ensures two adjacent vertices get different colors. Notice also that two adjacent vertices in the same set of Z will produce an unfeasible formula, as desired.

The formula can be constructed in $O(|V(G)|+|E(G)|)$ time and has $O(|V(G)|+|E(G)|)$ variables and clauses. Since the algorithm that solves 2-SAT is linear in the number of variables and clauses [5], we are done. □

We write $N(S)$ for the set of vertices of $V(G) \setminus S$ with a neighbor in S. For disjoint sets of vertices S, T of $V(G)$, let $N_T(S) = N(S) \cap T$. If $S = \{s\}$ we just write $N_T(s)$. For a vertex set S, let $\overline{S} = S \cup N(S)$. If $\overline{S} = V(G)$, we say that S is *dominating* G, or is a *dominating set*. Moreover, if S is dominating and the subgraph induced by S is connected, then we call S a *connected dominating set*. If $\overline{S}$ dominates G, we call S *2-dominating*.

For a graph G with a list system L, call a (nonempty) 2-dominating set $S \subseteq V(G)$ which induces a connected subgraph a *seed* of (G, L), if $|L(v)| = 1$ for each $v \in S$ and $|L(v)| = 2$ for each $v \in N(S)$. Note that we do not require the list system L to be updated.

Observe that for any seed S, and for any two non-adjacent vertices $v, w \in N(S)$ the following holds.

There is an induced v–w path of at least 3 vertices whose inner vertices all lie in S. (3.1)

The next result is essential to our proof.

3.1.4 (Camby and Schaudt [8]). *For all $t \geq 3$, any connected P_t-free graph has a connected dominating set whose induced subgraph is either P_{t-2}-free, or isomorphic to P_{t-2}.*

We use the following easy corollary of Theorem 3.1.4 in order to prove the existence of a small seed in P_7-free graphs that may be 3-colorable.

3.1.5. *Every connected P_7-free graph G has either a connected 2-dominating set of size at most 3 or a complete subgraph of 4 vertices. The set or the subgraph can be found in $O(|V(G)|^3|E(G)|)$ time.*

Proof. We prove the first statement by applying Theorem 3.1.4 to the graph in question, say G. Let S be the connected dominating set of G whose induced subgraph, say H, is either a P_5 or P_5-free. If H is a P_5, the three non-leaf vertices of H form a connected 2-dominating set of G, as desired. Otherwise, another application of Theorem 3.1.4 shows that H has a connected dominating set S' whose induced subgraph is either a P_3 or P_3-free. If $|S'| \leq 3$, S' is a connected 2-dominating set of G of at most three vertices. Otherwise, as a connected P_3-free graph is complete, $|S'| \geq 4$ implies that G contains a complete subgraph on 4 vertices.

Now we turn to the second statement. It suffices to run through all triples T of vertices ($O(|V(G)|^3)$ triples), and check if there is a common neighbor v of T such that $T \cup \{v\}$ induces a complete subgraph ($O(|E(G)|)$ possible vertices v). If not, we check whether T induces a connected subgraph and all vertices of the graph are within distance 2 from T. We can test the second property by using two steps of a breadth-first-search (that has time complexity $O(|E(G)|)$). □

This corollary will help us to reduce in the next section the original instance to a polynomial number of simpler instances. In each of these, the vertices having lists of size 1 or 2 satisfy some structural properties and the vertices having lists of size 3 form a stable set. We will in turn solve these special instances in Section 3.2.1 by reducing them to a polynomial number of instances to which we can apply Lemma 3.1.3.

3.2 Proof of the Main Theorem

Let G be a graph and v be a vertex of G. Observe first that if $G[N(v)]$ is not bipartite, then G is not 3-colorable. Observe also that if $G[N(v)]$ is a connected bipartite graph with bipartition U, W, then in every 3-coloring of G each of the sets U and W is monochromatic.

Let (G, L) be a list 3-coloring instance, such that for every $v \in V(G)$, $G[N(v)]$ is bipartite. We now describe a procedure that we call the *neighborhood reduction*.

If there is a vertex v with $|L(v)| = 3$ such that $G[N(v)]$ is connected, proceed as follows. Let U, W be a bipartition of $G[N(v)]$. We construct the graph G' we obtain from G by deleting v and replacing the neighborhood of v with an edge uw, where $N_{G'}(u) \cap V(G) = N_G(U) \cap V(G')$, and $N_{G'}(w) \cap V(G) = N_G(W) \cap V(G')$. In the case that W is empty, say, we can assume $U = \{u\}$, and we just define $G' = G - \{v\}$. The list of u is the intersection of all lists of vertices from U, and similar for w and W. Clearly, G admits a coloring for L if and only if G' admits a coloring for the new list system.

We iterate the above procedure until $G[N(v)]$ is disconnected for each vertex v with $|L(v)| = 3$. The term neighborhood reduction refers to the whole process until it stops.

The following claim says that this reduction preserves the property of being P_t-free, for $t \geq 3$.

(1) *If G is a P_t-free graph ($t \geq 3$), then the graph obtained from the neighborhood reduction is P_t-free.*

Proof. It suffices to consider one reduction step. Let us say we contracted the neighborhood of the vertex v in G, and obtained the graph G'. It remains to show that G' is still P_t-free.

To see this, suppose Q is an induced P_t in G'. Since G is P_t-free, it follows that $V(Q) \cap \{u, w\}$ is non-empty. Note that if Q contains both u and w, then u, w are consecutive on Q. So (in any case) we can write Q as $Q_1 - Q_2 - Q_3$, where $V(Q_2) \subseteq \{u, w\}$ and Q_1, Q_3 avoid $\{u, w\}$. We can assume that Q_1, Q_3 are not empty, as otherwise it is easy to substitute Q_2 with one or two vertices in $U \cup W$, and thus find an induced P_t in G, a contradiction.

Observe that Q_1, Q_3 each have exactly one vertex q_1, q_3 in $N(V(Q_2))$. If $|V(Q_2)| = 1$, we may assume both these vertices lie in $N(U)$, and we can substitute $Q_2 = u$ with either a common neighbor of q_1, q_3, or with a path $u_1 - v - u_2$ with $u_1 \in U \cap N(q_1)$ and $u_2 \in U \cap N(q_3)$. This gives an induced P_t in G, a contradiction.

So assume $|V(Q_2)| = 2$, and without loss of generality $Q_2 = u - w$, q_1 is adjacent to u and q_3 to w. Then q_1 is anticomplete to W and has a neighbor u_1 in U, and q_3 is anticomplete to U and has a neighbor w_1 in W. We can thus replace Q_2 with the path $u_1 - w_1$ if they are adjacent, or with the path $u_1 - v - w_1$ if they are not. This gives an induced P_t in G, yielding the final contradiction. □

Let G^* be a connected P_7-free graph with a list system L^*. We preprocess first the instance by applying the neighborhood reduction according to the input list system L^*, but, in order to simplify

the following presentation and discussion of our algorithm, after that preprocessing, we do not take the input list system L^* into account. Instead, we consider the list system L^1 with $L^1(v) = \{1, 2, 3\}$ for each vertex v. We intersect the current lists with L^* at the very end of the first phase of the algorithm only.

Here is an overview over the steps taken in the algorithm.

(a) Assert that for every vertex v of G^*, $G^*[N(v)]$ is bipartite. Otherwise, we can report that G^* is not 3-colorable.

(b) Reduce the instance so that the neighborhood of every vertex v with $|L^*(v)| = 3$ is disconnected. Let G be the graph obtained. By Claim 1, G is P_7-free.

(c) Apply Corollary 3.1.5 to G and obtain a connected 2-dominating set S_1 of size at most 3. (Notice that as we have asserted that every vertex has a bipartite neighborhood, G cannot contain a complete subgraph of size 4).

(d) For each feasible coloring of S_1 do the following to (G, L^1).

   (1) Update the lists of all remaining vertices to get a list system L^2 and a larger seed S_2. The set S_2 is the largest connected superset of S_1 containing only vertices with lists of size 1.

   (2) By guessing a partial coloring of the graph, obtain an equivalent set of list systems $\mathcal{L}_3$.

   (3) After another iteration, obtain a refined equivalent set of list systems $\mathcal{L}_4$.

   (4) For each list system $L \in \mathcal{L}_4$, intersect L with the input list system L^* and obtain a list system L'.

   (5) Update, and apply Lemma 3.2.1 to check for colorability.

We now describe the individual steps in more detail. The first step as well as the neighborhood reduction can be performed in $O(|V(G^*)|(|V(G^*)| + |E(G^*)|))$ time. The complexity associated to Corollary 3.1.5 is $O(|V(G)|^3|E(G)|)$ time. As we report that the graph is not 3-colorable otherwise, we may assume that $G[N(v)]$ is bipartite for every vertex v of G, and that we have obtained a 2-dominating connected set S_1 of G of size at most 3. For technical reasons, if S_1 is a singleton, we add one of its neighbors to S_1. Thus, we can assume that $|S_1| \geq 2$. We will go through all possible

3-colorings of S_1, and check for each whether it extends to a coloring of G which respects the list system L^*. This is clearly enough for deciding whether (G, L^*) is colorable.

So from now on, assume the coloring on S_1 is fixed and that for every other vertex v of G we have $L^1(v) = \{1, 2, 3\}$. We update the lists of all vertices in G. Note that updating can be done in $O(|V(G)| + |E(G)|)$ time, because each edge vw needs to be checked at most once (either updating v from w or updating w from v). After updating to list system L^2, consider the largest connected set S_2 of vertices with lists of size 1 that contains S_1. We claim that S_2 is a seed for (G, L^2). Indeed, since $\overline{S_1}$ dominates G, so does $\overline{S_2}$. Also, all vertices in $N(S_2)$ must have lists of size 2, since they are adjacent, but do not belong to S_2. So S_2 is a seed.

In the case that two adjacent vertices of S_2 have the same entry on their list, we abort the algorithm for that sub-instance and report that the current 3-coloring of S_1 does not lead to a valid 3-coloring of G.

(2) *For every vertex v in $N(S_2)$ there is an induced path on at least 3 vertices contained in $S_2 \cup \{v\}$ having v as an endpoint.*

Proof. This holds since S_2 is connected, $|S_2| \geq |S_1| \geq 2$, and v is not adjacent to two vertices of S_2 that have different entries on their lists (because $|L^2(v)| = 2$ after updating). □

Now, in two steps $j = 3, 4$, we will refine the set of subsystems of L^1 we are looking at, starting with $\mathcal{L}^2 = \{L^2\}$. At each step we replace the set $\mathcal{L}^{j-1}$ of list systems from the previous step with a set $\mathcal{L}^j$. More precisely, each element L of $\mathcal{L}^j$ is a subsystem of some element $Pred(L)$ of $\mathcal{L}^{j-1}$. We will argue below why it is sufficient to check colorability for the new set of list systems.

For each of the list systems L in $\mathcal{L}^j$, we will define a seed S_L and a set $T_L \subseteq N(S_L)$. We start with $S_{L^2} = S_2$ and T_{L^2} being the set of vertices $x \in N(S_{L^2})$ for which there does not exist an induced path $x - y - z$ with $|L^2(y)| = 3$ and $z \notin \overline{S_{L^2}}$. We will ensure for each list system L that $S_L \supseteq S_{Pred(L)}$ and $T_L \supseteq T_{Pred(L)}$. Furthermore, the seeds S_L and the sets T_L will have the following properties:

(A) for all $x \in N(S_L) \setminus T_L$, there is an induced path $x - y - z$ with $|L(y)| = 3$ and $z \notin \overline{S_L}$, and for no $x \in T_L$ is there such a path; and

(B) for each vertex $v \in V(G) \setminus \overline{S_L}$ either $|L(v)| = 1$ or $|L(v)| = 3$.

Let us now get into the details of the procedure. Successively, for $j = 3, 4$, we consider for each $L \in \mathcal{L}^{j-1}$ a set of subsystems of L obtained by partitioning the possible colorings of induced paths $x - y - z$ with $x \in N(S_L) \setminus T_L$, $|L(y)| = 3$ and $z \notin \overline{S_L}$ into a polynomial number of cases. The set $\mathcal{L}^j$ will be the union of all the sets of subsystems corresponding to lists L in $\mathcal{L}^{j-1}$. The idea is to make the seed grow, and after these two steps, obtain a set of list systems we can deal with, and such that the graph admits a coloring for the original list system if and only if it admits a coloring for one of the list systems in the set.

For each $i \in \{1, 2, 3\}$, let $\mathcal{P}_i$ be the set of paths $x - y - z$ with $x \in N(S_L) \setminus T_L$, $|L(y)| = 3$ and $z \notin \overline{S_L}$, and such that $i \notin L(x)$. We will order the paths of $\mathcal{P}_i$ non-increasingly by $|N(x) \setminus (N(y) \cup N(z) \cup \overline{S_L})|$, i.e., the number of vertices w (if any) such that $w - x - y - z$ is an induced path and $w \notin \overline{S_L}$.

We can compute and sort the paths of $\mathcal{P}_i$ in $O(|V(G)|^4)$ time. Moreover, this order of the paths induces an order on the set Y_i of vertices y that are midpoints of paths $x - y - z$ in $\mathcal{P}_i$. The vertices in Y_i are ordered by their first appearance as midpoints of the ordered paths in $\mathcal{P}_i$. Let $n_i = |Y_i|$, and $Y_i = \{y_{i,1}, \ldots, y_{i,n_i}\}$.

For each $i \in \{1, 2, 3\}$, we consider the following cases.

(a) All vertices in Y_i are colored i.

(b) There is a k, $1 \leq k \leq n_i$, such that the first $k - 1$ vertices of Y_i are colored i, and the first path $x - y_{i,k} - z$ in $\mathcal{P}_i$ is colored such that the color of $y_{i,k}$ is different from i, the color of every vertex in $W = N(x) \setminus (N(y_{i,k}) \cup N(z) \cup \overline{S_L})$ is i, and the color of z is i if W is empty.

(c) There is a k, $1 \leq k \leq n_i$, such that the first $k - 1$ vertices of Y_i are colored i, and the first path $x - y_{i,k} - z$ in $\mathcal{P}_i$ is colored such that the color of $y_{i,k}$ is different from i, the color of z is different from i if $W = N(x) \setminus (N(y_{i,k}) \cup N(z) \cup \overline{S_L})$ is empty, and if W is nonempty, there is a vertex w of W that gets a color different from i.

In order to do that, we consider all choices of functions $f : \{1, 2, 3\} \to \{a, b, c\}$. For each of these choices, we generate a set $\mathcal{L}_f$ of subsystems of L, and $\mathcal{L}^j$ will be the union of all sets $\mathcal{L}_f$. For fixed f the first step to obtain $\mathcal{L}_f$ consists of defining $\mathcal{L}_{i,f}$ for $i = 1, 2, 3$ in the following way.

If $\mathcal{P}_i$ is empty, then set $\mathcal{L}_{i,f} = \{L\}$. Otherwise, the set is as follows.

If $f(i) = a$, set $\hat{L}(y) = \{i\}$ for every $y \in Y_i$ and $\hat{L}(v) = L(v)$ for every $v \in V(G) \setminus Y_i$. Set $\mathcal{L}_{i,f} = \{\hat{L}\}$.

If $f(i) \neq a$, for each $k \in \{1, \ldots, n_i\}$, let x and z be such that $x - y_{i,k} - z$ is the first path in $\mathcal{P}_i$ having $y_{i,k}$ as midpoint, and let $W = N(x) \setminus (N(y_{i,k}) \cup N(z) \cup \overline{S_L})$.

If $f(i) = b$, consider all subsystems $\hat{L}$ of L which only differ from L on $W \cup \{y_{i,1}, \ldots, y_{i,k}, z\}$, and satisfy $\hat{L}(y_{i,k}) = \{i'\}$ for some $i' \neq i$, $\hat{L}(v) = \{i\}$ for all $v \in W \cup \{y_{i,1}, \ldots, y_{i,k-1}\}$, $|\hat{L}(z)| = 1$, and $\hat{L}(z) = \{i\}$ if W is empty. Update these list systems $\hat{L}$ from $y_{i,k}$ except on T_L and let $\mathcal{L}_{i,f}$ be the set of all list systems found in this way, for every choice of k. Note that, in each list system, the updated list of x has size 1, and that the number of list systems generated this way is $O(|V(G)|)$.

If $f(i) = c$, if W is nonempty, for each $w \in W$ consider all subsystems $\hat{L}$ of L which only differ from L on $\{y_{i,1}, \ldots, y_{i,k}, z, w\}$, and satisfy $\hat{L}(v) = \{i\}$ for all $v \in \{y_{i,1}, \ldots, y_{i,k-1}\}$, $|\hat{L}(y_{i,k})| = |\hat{L}(z)| = |\hat{L}(w)| = 1$, $\hat{L}(y_{i,k}) \neq \{i\}$, and $\hat{L}(w) \neq \{i\}$. If W is empty, consider all subsystems $\hat{L}$ of L which only differ from L on $\{y_{i,1}, \ldots, y_{i,k}, z\}$, and satisfy $\hat{L}(v) = \{i\}$ for $v \in \{y_{i,1}, \ldots, y_{i,k-1}\}$, $|\hat{L}(y_{i,k})| = |\hat{L}(z)| = 1$, $\hat{L}(y_{i,k}) \neq \{i\}$, and $\hat{L}(z) \neq \{i\}$. Update these list systems $\hat{L}$ from $y_{i,k}$ except on T_L and let $\mathcal{L}_{i,f}$ be the set of all list systems found in this way, for every choice of k and of w (if such a w exists). Note that again, in each list system, the updated list of x has size 1, and that the number of list systems generated this way is $O(|V(G)|^2)$.

Finally, for each triple $(L_1, L_2, L_3) \in \mathcal{L}_{1,f} \times \mathcal{L}_{2,f} \times \mathcal{L}_{3,f}$ consider the list system $\tilde{L}$ obtained from intersecting the lists of L_1, L_2, L_3, taking intersections at each vertex. Update the list system $\tilde{L}$ from any vertex in S_L, except on T_L. Let $\mathcal{L}_f$ be the set of all list systems $\tilde{L}$ thus generated.

Observe that $|\mathcal{L}_f| = O(|V(G)|^6)$, since $|\mathcal{L}_{i,f}| = O(|V(G)|^2)$ for $i = 1, 2, 3$.

For each $L' \in \mathcal{L}_f$, let $S_{L'}$ be a maximal connected set of vertices with list size 1 that contains S_L. Then $S_{L'}$ is a seed.

Note that for each $L' \in \mathcal{L}_f$, all vertices v in T_L satisfy $|L'(v)| = 2$, since they were never updated. Let $T_{L'}$ be the union of T_L with all vertices $x \in N(S_{L'})$ which are not the starting point of an induced path $x - y - z$ with $|L'(y)| = 3$ and $z \notin \overline{S_{L'}}$.

Clearly, $T_{L'} \subseteq N(S_{L'})$. Property (A) holds because of the way we defined $T_{L'}$, and because there are no new paths of the type described in (A) that start at vertices in T_L, as seeds grow and lists shrink. Property (B) holds because S_L was a seed satisfying Properties (A) and (B), and when defining list systems in $\mathcal{L}_f$ by the cases (a), (b), and (c), we have reduced the size of some vertex

lists from 3 to 1, never to 2; then we only updated *from vertices* in S_L except on T_L, thus every vertex that got a list of size 1 by updating is connected to S_L by a path all whose vertices have current lists of size one, and is now in $S_{L'}$ and, consequently, every vertex that got a list of size 2 by updating is in $N(S_{L'})$.

(3) *There is a coloring of G for the list system L^2 if and only if G has a coloring for at least one of the list systems in $\mathcal{L}^4$.*

Proof. Indeed, observe that when obtaining $\mathcal{L}^j$ from $\mathcal{L}^{j-1}$, we consider for each $L \in \mathcal{L}^{j-1}$ and for each $i \in \{1,2,3\}$ the possibility that all induced 3-vertex-paths that start in $N(S_L)$ and then leave $\overline{S_L}$ have their second vertex colored i (when $f(i) = a$). We also consider the possibility that there is such a path whose second vertex is colored with a different color (when $f(i) = b$ or $f(i) = c$). In that case, we consider separately the possible colorings of a fourth vertex w, if such a w exists. □

Note that $|\mathcal{L}^{j+1}| = O(|\mathcal{L}^j| \cdot |V(G)|^6)$ for each $j = 2, 3$. Since $|\mathcal{L}^2| = 1$, the number of list systems in $\mathcal{L}^4$ is $O(|V(G)|^{12})$.

Next we prove that during the above described process, the union of our seed with the set T_L actually grows.

(4) *For each $L \in \mathcal{L}^j$, we have $N(S_{Pred(L)}) \subset S_L \cup T_L$.*

Proof. Let $L' = Pred(L)$ and let f be the function used to produce L from L'. In order to see Claim 4, suppose there is a vertex $x' \in N(S_{L'}) \setminus (S_L \cup T_L)$. As $x' \notin S_L$ and $S_L \supseteq S_{L'}$, we know that $x' \in N(S_L)$. Furthermore, since $x' \notin T_L$, there is an induced path $x' - y' - z'$ with $|L(y')| = 3$ and $z' \notin \overline{S_L}$. In particular, $z' \notin \overline{S_{L'}}$ and since lists only shrink, $|L'(y')| = 3$. So $f(i) \neq a$, where i is such that $i \notin L(x') = L'(x')$. Thus $f(i) \in \{b, c\}$, and so there is an induced path $x - y - z$ with $x \in N(S_{L'})$, $y, z \notin N(S_{L'})$, $L(x) \neq \{i\}$, $L(y) \neq \{i\}$, and $|L(x)| = |L(y)| = |L(z)| = 1$, and thus $x, y, z \in S_L$. Since $y', z' \notin \overline{S_L}$, it follows that there are no edges between $\{y', z'\}$ and $\{x, y, z\}$. Also, since $x' \in N(S_L)$, there are no edges from x' to vertices $v \in \{x, y, z\}$ with $L(v) \subsetneq L(x')$. In other words, the only possible edge between $\{x, y, z\}$ and $\{x', y', z'\}$ is $x'z$, and if this edge is present, we have that $L(z) = \{i\}$. On the other hand, by (3.1), there is a path Q of at least 3 vertices connecting x and x' whose interior lies in $S_{L'}$ (in particular, the interior of Q is anticomplete to $\{y', z', y, z\}$). So, since G is P_7-free, the edge $x'z$ has to be present and thus we have $L(z) = \{i\}$.

Now, assume there is an extension of $x-y-z$ to an induced path $w-x-y-z$ with $w \notin \overline{S_{L'}}$. Then, as the sequence $w-x-y-z-x'-y'-z'$ is not an induced P_7, there is an edge from w to one of x', y', z'. Observe if $|L(w)| = 1$, then $w \in S_L$ and neither wy' nor wz' is an edge. Hence either $|L(w)| \geq 2$, or $L(w) = \{i\}$, and in the latter case the only edge from w to $\{x', y', z'\}$ is wx'. As this happens for all possible choices of w, we see that $f(i) \neq c$, and thus $f(i) = b$. This means that for all possible w, w is adjacent to x'. But now, observe that

$$N(x) \setminus (N(y) \cup N(z) \cup \overline{S_{L'}}) \subsetneq N(x') \setminus (N(y') \cup N(z') \cup \overline{S_{L'}}),$$

since z is in the right hand side set, but not in the left hand side set. This is a contradiction to the choice of the path $x-y-z$ for the definition of L from L' and f.

We conclude that there is no extension of $x-y-z$ to an induced path $w-x-y-z$. But then, the fact that $L(z) = \{i\}$ implies that again, $f(i) \neq c$, and thus, $f(i) = b$. The existence of the edge $x'z$ gives a contradiction to the choice of the path $x-y-z$ for the definition of L from L' and f. This proves Claim 4. □

Next, we prove that two steps of performing the above procedure suffice to take care of all paths on three vertices that start in the boundary of the current seed, and then leave the seed.

(5) *For each $L \in \mathcal{L}^4$, we have $N(S_L) \subset T_L$.*

Proof. Suppose there are $L \in \mathcal{L}^4$ and $x \in N(S_L)$ such that $x \notin T_L$. Then by (A) there is a path $x-y-z$ with $|L(y)| = 3$ and $z \notin \overline{S_L}$. Clearly y and z are anticomplete to S_L. Let $L' = Pred(L)$ and $L'' = Pred(L')$. Choose an induced path P from x to $N(S_{L''})$ with all vertices but x in S_L, say it ends in $x'' \in N(S_{L''})$. By Claim 4, $N(S_{L''}) \subseteq S_{L'} \cup T_{L'}$. On the other hand, as $T_{L'} \cap S_L = \emptyset$, $x'' \in S_{L'}$. In particular, $x \neq x''$.

Let x_1 be the neighbor of x in P. Since $x \notin S_L \cup T_L$, by Claim 4, $x_1 \in S_L \setminus S_{L'}$. As the subpath of P from x_1 to x'' goes from $S_L \setminus S_{L'}$ to $S_{L'}$, it contains a vertex x' in $N(S_{L'})$. The vertex x' may be x_1, but x' is different from x'' because $x'' \in S_{L'}$. As x' is in the subpath from x_1 to x'', $x' \neq x$. Summing up, x, x' and x'' are three distinct vertices, and so P together with the path $x-y-z$ and the path provided by Claim 2 for x'' gives a path on at least 7 vertices, a contradiction. □

By Claim 3, (G, L^2) is colorable if and only if (G, L) is colorable for some $L \in \mathcal{L}^4$. For each $L \in \mathcal{L}^4$ our aim is to check whether there is a coloring of (G, L). This we will do, after some more

discussion, with the help of Lemma 3.2.1 below. So from now on, let $L \in \mathcal{L}^4$ be fixed. Let X be the set of all vertices in $V(G) \setminus \overline{S_L}$ with lists of size 1, and set $Y = V(G) \setminus (\overline{S_L} \cup X)$. By construction, $|L(y)| = 3$ for each $y \in Y$.

By Claim 5, no vertex of $N(S_L)$ is the starting point of an induced path $x - y - z$ with $y \in Y$ and $z \in X \cup Y$. In other words, for each $y \in Y$, all edges between $N(y) \cap \overline{S_L}$ and $N(y) \setminus \overline{S_L}$ are present.

Now we intersect L with the given input list system L^*, and then update. Let L' be the resulting list system. We may assume that $|L'(v)| \geq 1$ for all $v \in V(G)$, otherwise we may safely report that (G, L') is not colorable, and thus L does not lead to a feasible coloring of (G, L^*). Let Y' be the set of vertices y of Y such that $|L'(y)| = 3$. We noticed that for each $y \in Y$, all the edges between $N(y) \cap \overline{S_L}$ and $N(y) \setminus \overline{S_L}$ are present. Since Y' is a subset of the vertices v such that $|L^*(v)| = 3$ and we have applied the neighborhood reduction at the beginning of the algorithm and the graph did not change, for $y \in Y'$ one of the sets $N(y) \cap \overline{S_L}$ or $N(y) \setminus \overline{S_L}$ must be empty. Since $\overline{S_L} \supseteq \overline{S_2}$ is a dominating set, we conclude that $N(y) \setminus \overline{S_L} = \emptyset$, and thus

$$N(y) \subseteq \overline{S_L} \text{ for each } y \in Y'. \tag{3.2}$$

Consider the set S' of all vertices that are connected to S_L by a (possibly trivial) path containing only vertices with lists L' of size 1. Note that S' is a seed. In particular, $S_L \subseteq S'$ and by (3.2), we have $N(y) \subseteq \overline{S'}$ for every $y \in Y'$. That is, Y' is a stable set anticomplete to $V(G) \setminus (\overline{S'} \cup Y')$.

We are now in a situation where the following lemma applies, solving the remaining problem.

3.2.1. *Let G be a connected P_7-free graph with a list system L. Let S be a seed of G such that if $v \in S$ and $w \in N(S)$ are adjacent, then they do not share list entries. Assume that the set X of vertices having lists of size 3 is stable and anticomplete to $V(G) \setminus (\overline{S} \cup X)$. Assume also that no vertex in X has a connected neighborhood. Then we can decide whether G has a coloring for L in $O(|V(G)|^9(|V(G)| + |E(G)|))$ time.*

The next subsection is devoted to the proof of Lemma 3.2.1. Since we have $|\mathcal{L}^4| = O(|V(G)|^{12})$ many lists to consider, and need to apply Lemma 3.2.1 to each of these, the total running time of the whole algorithm amounts to $O(|V(G)|^{21}(|V(G)| + |E(G)|))$.

3.2.1 Proof of Lemma 3.2.1

Let G, L, S and X be as in the statement of Lemma 3.2.1.

In this proof we make extensive use of the concept of monochromatic set constraints as defined in Section 3.1. Note that (G, L) is colorable if and only if the triple $(G, L, \emptyset)$ is colorable. Our aim is to define a set $\mathcal{R}$ of restrictions of $(G, L, \emptyset)$ with the property that in any element of $\mathcal{R}$ there are no vertices with list of size 3, and $(G, L, \emptyset)$ is colorable if and only if $\mathcal{R}$ is colorable. Moreover, $\mathcal{R}$ has polynomial size and is computable in polynomial time.

If $X = \emptyset$, we simply let $\mathcal{R} = \{(G, L, \emptyset)\}$. Otherwise, for all $i = 1, 2, 3$, let D_i be the set of vertices $v \in N(S)$ with $L(v) = \{1, 2, 3\} \setminus \{i\}$, and for $x \in X$, let $N_i(x) = N(x) \cap D_i$, for $i = 1, 2, 3$. Observe that, under the hypothesis of Lemma 3.2.1, for every $d \in D_i$ and for every $s \in S \cap N(d)$, we have $L(s) = \{i\}$. By the same hypothesis, no vertex of X has neighbors in S.

If $N(x) \subseteq D_i$ for some $x \in X$ and some $i \in \{1, 2, 3\}$, then setting $L(x) = \{i\}$ does not change the colorability of $(G, L, \emptyset)$, so we may assume that for every $x \in X$ at least two of the sets $N_1(x), N_2(x), N_3(x)$ are non-empty. Let X_1 be the set of vertices $x \in X$ for which $N_2(x)$ is not complete to $N_3(x)$; for every $x \in X_1$ fix two vertices $n_2(x) \in N_2(x)$ and $n_3(x) \in N_3(x)$ such that $n_2(x)$ is non-adjacent to $n_3(x)$. Define similarly X_2 and $n_1(x), n_3(x)$ for every $x \in X_2$, and X_3 and $n_1(x), n_2(x)$ for every $x \in X_3$. Since no vertex of X has a connected neighborhood and X is a stable set and anticomplete to $V(G) \setminus (\overline{S} \cup X)$, it follows that $X = X_1 \cup X_2 \cup X_3$.

Before we state the coloring algorithm, we need some auxiliary statements. For a path P with ends u, v let $P^* = V(P) \setminus \{u, v\}$ denote the interior vertices of P.

(1) *Let $i, j \in \{1, 2, 3\}$, $i \neq j$, and let $u_i, v_i \in D_i$ and $u_j, v_j \in D_j$, such that $\{u_i, v_i, u_j, v_j\}$ is a stable set. Then there exists an induced path P with ends $a, b \in \{u_i, v_i, u_j, v_j\}$ such that*

(a) $\{a, b\} \neq \{u_i, u_j\}$ and $\{a, b\} \neq \{v_i, v_j\}$,

(b) P^* is contained in S and, in particular, $|L(v)| = 1$ for every $v \in P^*$, and

(c) P^* is anticomplete to $\{u_i, v_i, u_j, v_j\} \setminus \{a, b\}$.

Proof. Note that each of u_i, u_j, v_i, v_j has a neighbor in S, and $G[S]$ is connected. Let P be an induced path with $P^* \subseteq S$ that connects u_i with v_i. If P is not as desired, at least one of u_j, v_j has a neighbor on P. Let p be the neighbor of u_j or v_j on P that is closest to v_i; by symmetry we may assume p is a neighbor of u_j. Note that p is not adjacent to $u_i, v_i \in D_i$, because p is already

adjacent to $u_j \in D_j$. Hence, if $u_j - p - P - v_i$ is not as desired, then v_j must have a neighbor on $p - P - v_i$. Among all such neighbors, let p' be the one that is closest to p (possibly $p' = p$). As before, p' is not adjacent to any of $u_i, v_i \in D_i$, and thus, $u_j - p - P - p' - v_j$ is the desired path. □

(2) *Let* $\{i, j, k\} = \{1, 2, 3\}$. *Let* $x, y \in X_i$, *let* $n_j \in N_j(x)$ *and* $n_k \in N_k(x)$ *such that* n_j *is non-adjacent to* n_k. *Then there is an edge between* $\{x, n_j, n_k\}$ *and* $\{y, n_j(y), n_k(y)\}$.

Proof. Assume there is no such edge. Then in particular, vertices $n_j, n_j(y), n_k, n_k(y)$ are distinct, and we can apply Claim 1 to obtain a path P with $P^* \subseteq S$ that connects two vertices from $\{n_j, n_j(y), n_k, n_k(y)\}$ in way that P^*, together with $n_j - x - n_k$ and $n_j(y) - y - n_k(y)$, forms an induced path of length at least 7, a contradiction. □

Next we distinguish between several types of colorings of G, and show how to reduce the list sizes assuming that a coloring of a certain type exists. For this, let $\{i, j, k\} = \{1, 2, 3\}$. We call a coloring c of a restriction (G', L', Z') of (G, L, Z)

(A) a *type A coloring with respect to* i if there exists an induced path $n_j - x - n_k - z - m_j$ with $x, z \in X_i$, $n_j \in N_j(x)$, $m_j \in N_j(z)$, and $n_k \in N_k(x) \cap N_k(z)$ such that $c(n_j) = i$, $c(x) = j$ and $c(z) = k$ (this implies $c(n_k) = c(m_j) = i$), or the same with the roles of j and k reversed;

(B) a *type B coloring with respect to* i if it is not a type A coloring with respect to i, and there exists an induced path $x - n_k - z - m_j$ with $x, z \in X_i \cap V(G')$, $m_j \in N_j(z)$, $n_k \in N_k(x) \cap N_k(z)$ such that $c(x) = j$ and $c(z) = k$ (this implies $c(n_k) = c(m_j) = i$), or the same with the roles of j and k reversed;

(C) a *type C coloring with respect to* i if it is not a type A or type B coloring, and there exist $z \in X_i \cap V(G')$, $m_j \in N_j(z)$ and $n_k \in N_k(z)$ such that $c(m_j) = c(n_k) = i$.

We will show in Claim 3 how to refine the instances to test if a graph admits a type A coloring with respect to a color i; in Claim 4 how to refine the instances to test if a graph admits a type B coloring with respect to i under the assumption that it does not admit a type A coloring with respect to i; in Claim 5 how to refine the instances to test if a graph admits a type C coloring with respect to i under the assumption that it does not admit a type A or type B coloring with respect to i; finally, in Claim 6 we show how to refine the instances to test if a graph admits a coloring under

the assumption that it does not admit a type A, or type B, or type C coloring with respect to i. After the claims, we describe how to combine them in order to obtain the desired list of restrictions of the original instance.

(3) *Let (G', L', Z') be a restriction of (G, L, Z). There exists a set $\mathcal{L}_i$ of $O(|V(G)|^3)$ subsystems of L' such that*
(a) $|L''(v)| \leq 2$ for every $L'' \in \mathcal{L}_i$ and $v \in X_i \cap V(G')$, and
(b) (G', L', Z') admits a type A coloring with respect to i if and only if $(G', \mathcal{L}_i, Z')$ is colorable.
Moreover, $\mathcal{L}_i$ can be constructed in $O(|V(G)|^4)$ time.

Proof. For every $x, z \in X_i \cap V(G')$ and $n_j \in N_j(x)$ for which there are $n_k \in N_k(x) \cap N_k(z)$ and $m_j \in N_j(z)$ such that $n_j - x - n_k - z - m_j$ is an induced path, we construct a list system $L'' = L_{x,z,n_j}$ depending on x, z, n_j; for the same case with triples $x, z \in X_i \cap V(G')$, $n_k \in N_k(x)$, and the roles of j and k reversed, we construct in an analogous way a list system $L'' = L'_{x,z,n_k}$ depending on x, z, n_k. The set $\mathcal{L}_i$ will be the set of all list systems L'' obtained in this way. So the number of list systems in $\mathcal{L}_i$ is $O(|V(G)|^3)$.

For x, z, n_j as above (we will assume the first case in the definition, the other case is analogous), we define L'' by setting $L''(x) = \{j\}$, $L''(z) = \{k\}$, $L''(n_j) = \{i\}$, and leaving $L''(v) = L'(v)$ for all $v \in V(G') \setminus \{x, z, n_j\}$. Update $N_j(z)$ from z, and $N_k(x)$ from x. Let n_k and m_j be such that $n_k \in N_k(x) \cap N_k(z)$, $m_j \in N_j(z)$, and $n_j - x - n_k - z - m_j$ is an induced path. Note that after updating, $L''(n_k) = L''(m_j) = \{i\}$. Now, for each vertex $v \in D_j \cup D_k$ that has a neighbor $v' \in \{x, z, n_j, n_k, m_j\}$, update v from each such neighbor v'. Next, for every vertex $y \in X_i \cap V(G')$, if $n_j(y)$ or $n_k(y)$ now has list size 1, then update y from both $n_j(y)$ and $n_k(y)$, and also update y from m_j, n_j and n_k in the case that y is adjacent to any of them. Call the obtained list system L'' (slightly abusing notation). By the way we updated, it only takes $O(|V(G)|)$ time to compute this list system. The total time for constructing all list systems for $\mathcal{L}_i$ thus amounts to $O(|V(G)|^4)$.

In order to see Claim 3 (a), we need to show that $|L''(y)| \leq 2$ for all $y \in X_i \cap V(G')$. For contradiction, suppose $|L''(y)| = 3$ for some $y \in X_i \cap V(G')$. By Claim 2, there must be edges between $\{x, n_j, n_k\}$ and $\{y, n_j(y), n_k(y)\}$, and also between $\{z, m_j, n_k\}$ and $\{y, n_j(y), n_k(y)\}$. By the way we updated L'', the only possibly edges between these sets are those connecting $n_j(y)$ with x, and $n_k(y)$ with z. Consequently, $n_j(y)x$ and $n_k(y)z$ are both edges, and so $m_j - z - n_k(y) - y - n_j(y) - x - n_j$ is a P_7, a contradiction.

For Claim 3 (b), first note that by construction, if $(G', \mathcal{L}_i, Z')$ is colorable then (G', L', Z') has a type A coloring with respect to i. On the other hand, if c is a type A coloring of (G', L', Z') with respect to i, then there is an induced path $n_j - x - n_k - z - m_j$ with $x, z \in X_i$, $n_j, m_j \in N_j(x)$, and $n_k \in N_k(x)$ such that $c(n_j) = c(m_j) = c(n_k) = i$, $c(x) = j$, and $c(z) = k$ (or the same with the roles of j and k reversed). Since updating does not change the set of possible colorings for a list, c is a coloring for the list $L'' = L_{x,z,n_j}$ (respectively, $L'' = L_{x,z,n_k}$). So $\mathcal{L}_i$ is as required for Claim 3 (b). □

(4) *Let (G', L', Z') be a restriction of (G, L, Z) that does not admit a type A coloring. There exists a set $\mathcal{L}_i$ of $O(|V(G)|^2)$ subsystems of L' such that*
(a) $|L''(v)| \leq 2$ for every $L'' \in \mathcal{L}_i$ and $v \in X_i \cap V(G')$, and
(b) (G', L', Z') admits a type B coloring with respect to i if and only if $(G', \mathcal{L}_i, Z')$ is colorable.
Moreover, $\mathcal{L}_i$ can be constructed in $O(|V(G)|^3)$ time.

Proof. For every $x, z \in X_i \cap V(G')$ for which there exist $n_k \in N_k(x) \cap N_k(z)$ and $m_j \in N_j(z)$ such that $x - n_k - z - m_j$ is an induced path, we construct a list system $L'' = L_{x,z}$, depending on x and z. For the case with the roles of j and k reversed, we construct analogously a list system $L'' = L'_{x,z}$. The set $\mathcal{L}_i$ will be the set of all list systems L'' obtained in this way. So the number of list systems in $\mathcal{L}_i$ is $O(|V(G)|^2)$.

Given a pair of vertices x, z in $X_i \cap V(G')$ satisfying the hypothesis, let n_k and m_j such that $n_k \in N_k(x) \cap N_k(z)$, $m_j \in N_j(z)$, and $x - n_k - z - m_j$ is an induced path. Let M be the set of all $n \in N_j(x)$ for which $n - x - n_k - z - m_j$ is an induced path.

Define L'' by setting $L''(x) = \{j\}$, $L''(z) = \{k\}$, $L''(n_k) = L''(m_j) = \{i\}$, and $L''(n) = \{k\}$ for all $n \in M$, and leaving $L''(v) = L'(v)$ for all $v \in V(G') \setminus (\{x, z, n_k, m_j\} \cup M)$. Now, for each vertex $v \in D_j \cup D_k$ that has a neighbor v' in $\{x, z, m_j, n_k\}$, update v from each such neighbor v'. Next, for every vertex $y \in X_i \cap V(G')$, if $n_j(y)$ or $n_k(y)$ now has list size 1, then update y from both $n_j(y)$ and $n_k(y)$, and also update y from m_j and n_k in the case that y is adjacent to either of them. Call the obtained list system L''. Note that by the way we updated, it takes $O(|V(G)|)$ time to compute this list system. The total time for constructing all list systems for $\mathcal{L}_i$ thus amounts to $O(|V(G)|^3)$.

In order to see Claim 4 (a), we need to show that $|L''(y)| \leq 2$ for all $y \in X_i \cap V(G')$. For contradiction, suppose $|L''(y)| = 3$ for some $y \in X_i \cap V(G')$. Then $n_j(y) \notin M \cup \{m_j\}$ and

$n_k(y) \neq n_k$. By Claim 2, it follows that $n_k(y)$ is adjacent to z, and by the way we updated L'', the only other possible edge between $\{x, n_k, z, m_j\}$ and $\{y, n_j(y), n_k(y)\}$ would be $xn_j(y)$. However, since $n_j(y) \notin M$, we deduce that $n_j(y)$ is non-adjacent to x. Let s be a neighbor of $n_j(y)$ in S with $L(s) = \{j\}$. Then s is anticomplete to $\{n_k, x, y, z, n_k(y)\}$. So $x - n_k - z - n_k(y) - y - n_j(y) - s$ is a P_7, a contradiction.

For Claim 4 (b), note that by construction, if $(G', \mathcal{L}_i, Z')$ is colorable then (G', L', Z') has a type B coloring with respect to i. On the other hand, if c is a type B coloring of (G', L', Z') with respect to i, then there is an induced path $x - n_k - z - m_j$ with $x, z \in X_i$, $m_j \in N_j(x)$, and $n_k \in N_k(x) \cap N_k(z)$ such that $c(m_j) = c(n_k) = i$, $c(x) = j$, and $c(z) = k$ (or the same with the roles of j and k reversed). Since c is not a type A coloring, it follows that $c(v) = k$ for all v in M. Since updating does not change the set of possible colorings for a list, c is a coloring for $L'' = L_{x,z}$. So $\mathcal{L}_i$ is as required for Claim 4 (b). □

(5) *Let (G', L', Z') be a restriction of (G, L, Z) that does not admit a type A or type B coloring. There exists a set $\mathcal{L}_i$ of $O(|V(G)|^2)$ subsystems of L' such that*
(a) $|L''(v)| \leq 2$ for every $L'' \in \mathcal{L}_i$ and $v \in X_i \cap V(G')$, and
(b) (G', L', Z') admits a type C coloring with respect to i if and only if $(G', \mathcal{L}_i, Z')$ is colorable.
Moreover, $\mathcal{L}_i$ can be constructed in $O(|V(G)|^4)$ time.

Proof. For every $z \in X_i \cap V(G')$ having non-adjacent neighbors $m_j \in N_j(z)$ and $n_k \in N_k(z)$, we construct two families of list systems, one for each of the possible colors j, k of z in a type C coloring, z, m_j, n_k are as in the definition of a type C coloring. We only describe how to obtain the family of list systems L'' with $L''(z) = \{k\}$; the definition of the family of list systems L'' with $L''(z) = \{j\}$ is analogous, with the roles of j and k reversed.

Let N_z be the set of vertices n_k in $N_k(z)$ having a non-neighbor in $N_j(z)$. For each such vertex n_k, let $W = W_{z,n_k}$ be the set of all $w \in X_i \cap V(G')$ such that there exists an induced path $w - n_k - z - m_j$ with $m_j \in N_j(z)$. We will order the vertices of N_z non-increasingly by $|W|$. We can compute and sort the vertices of N_z in $O(|V(G)|^3)$ time.

For each $n_k \in N_z$, define $L'' = L_{z,n_k}$ by setting $L''(z) = L''(w) = \{k\}$ for all $w \in W$, $L''(n_k) = \{i\}$, $L''(n'_k) = \{j\}$ for every $n'_k \in N_z$ having an index lower than the index of n_k in N_z, and leaving $L''(v) = L'(v)$ for all the remaining vertices. Update each vertex of $N_j(z)$ from z. Now, for each

vertex v that has a neighbor in $\{z\} \cup N_k(z) \cup N_j(z) \cup W$, update v from each such neighbor v'. Next, for every vertex $y \in X_i \cap V(G')$, if $n_j(y)$ or $n_k(y)$ now has list size 1, then update y from both $n_j(y)$ and $n_k(y)$. Call the obtained list system L''. Note that by the way we updated, it takes $O(|V(G)|^2)$ time to compute this list system. The number of list systems L_{z,n_k} is $O(|V(G)|^2)$, and the same for the case with the roles of j and k reversed. Then $\mathcal{L}_i$, the set of all list systems obtained in this way, has cardinality $O(|V(G)|^2)$, and can be constructed in $O(|V(G)|^4)$ time. We may assume that $|L''(v)| \geq 1$ for all $v \in V(G')$, otherwise we detect that the list system L'' does not lead to a feasible solution to L'.

In order to see Claim 5 (a), we need to show that $|L''(y)| \leq 2$ for all $y \in X_i \cap V(G')$. For contradiction, suppose $|L''(y)| = 3$ for some $y \in X_i \cap V(G')$. Let m_j be a non-neighbor of n_k in $N_j(z)$. Note that by the way we updated, $L''(m_j) = \{i\}$. Claim 2 guarantees an edge between $\{z, m_j, n_k\}$ and $\{y, n_j(y), n_k(y)\}$. By the way we updated L'', $n_j(y) \neq m_j$, $n_k(y) \neq n_k$, z is not adjacent to $n_j(y)$, and there is no edge between $\{m_j, n_k\}$ and $\{y, n_j(y), n_k(y)\}$. So z is adjacent to $n_k(y)$. Since $n_k(y)$ is not adjacent to m_j, $n_k(y)$ belongs to N_z, and as it has two colors in its list L'', its index is greater than the index of n_k in N_z. As y is adjacent to $n_k(y)$ and not to $\{m_j, n_k\}$, $y \in W_{z,n_k(y)} \setminus W_{z,n_k}$. Since $|W_{z,n_k}| \geq |W_{z,n_k(y)}|$, there is a vertex $x \in W_{z,n_k} \setminus W_{z,n_k(y)}$. By definition, $L''(x) = \{k\}$, thus x is not adjacent to $\{y, n_j(y)\}$. Let s be a neighbor of $n_j(y)$ in S with $L(s) = \{j\}$. Then s is anticomplete to $\{n_k, x, y, z, n_k(y)\}$. So $x - n_k - z - n_k(y) - y - n_j(y) - s$ is a P_7, a contradiction.

For Claim 5 (b), note that by construction, if $(G', \mathcal{L}_i, Z')$ is colorable then (G', L', Z') has a type C coloring with respect to i. On the other hand, if c is a type C coloring of (G', L', Z') with respect to i, then there is a path $n_k - z - m_j$ with $z \in X_i \cap V(G')$, $m_j \in N_j(z)$, $n_k \in N_k(z)$, and $c(m_j) = c(n_k) = i$. Assume $c(z) = k$ (the case $c(z) = j$ is analogous), and consider the path $n_k - z - m_j$ that minimizes the index of n_k in N_z. Since $c(m'_j) = i$ for every m'_j in $N_j(z)$, it follows that $c(n'_k) = j$ for every $n'_k \in N_z$ having a lower index than the index of n_k in N_z.

Since c is not a type A or B coloring, for every vertex $w \in W_{z,n_k}$ we have $c(w) = c(z) = k$. Since updating does not change the set of possible colorings for a list, c satisfies the list system $L'' = L_{z,n_k}$. So $\mathcal{L}_i$ is as required for Claim 5 (b). □

(6) *Let (G', L', Z') be a restriction of (G, L, Z). Assume that (G', L', Z') does not admit a type A, type B, or type C coloring with respect to i (i.e., no coloring with a vertex x of $X_i \cap V(G')$ having*

neighbors colored i both in $N_j(x)$ and $N_k(x)$). Let Y_i be the set of vertices $x \in X_i \cap V(G')$ such that $N_i(x) = \emptyset$, and let $Z_i = \bigcup_{y \in Y_i}\{N_j(y), N_k(y)\}$. Then (G', L', Z') is colorable if and only if $(G' \setminus Y_i, L', Z' \cup Z_i)$ is colorable, and any 3-coloring of $(G' \setminus Y_i, L', Z' \cup Z_i)$ can be extended to a 3-coloring of (G', L', Z') in $O(|V(G)|)$ time.

Proof. It is enough to prove that for every coloring c of (G', L', Z') and every $x \in X_i \cap V(G')$ such that $N_i(x) = \emptyset$, the sets $N_j(x)$ and $N_k(x)$ are monochromatic with respect to c. Supposing this is false, we may assume that for some coloring c there are vertices $u, v \in N_j(x)$ with $c(u) = i$ and $c(v) = k$. Since there are no type A or type B colorings and c is not of type C, it follows that $c(w) = j$ for every $w \in N_k(x)$. But then x has neighbors of all three colors, contrary to the fact that c is a coloring. □

Let $Z = \emptyset$. Recall that our aim was to define a set $\mathcal{R}$ of restrictions of (G, L, Z) with the property that in any element of $\mathcal{R}$ there are no vertices with list of size 3, and such that (G, L, Z) is colorable if and only if $\mathcal{R}$ is colorable. We now construct $\mathcal{R}$ as follows. Apply Claims 3, 4, 5 and 6 with $i = 1$ to (G, L, Z) to create sets $\mathcal{R}_2, \ldots, \mathcal{R}_5$, each consisting of $O(|V(G)|^3)$ restrictions of (G, L, Z). For every $x \in X_1$ and every $(G', L', Z') \in \mathcal{R}_2 \cup \mathcal{R}_3 \cup \mathcal{R}_4$, we have that $|L'(x)| \leq 2$. For $(G', L', Z') \in \mathcal{R}_5$, if $x \in X_1$ and $|L'(x)| = 3$, then $N_j(x) \neq \emptyset$ for every $j \in \{1, 2, 3\}$. Repeat this with $i = 2$ for every restriction in $\mathcal{R}_2 \cup \mathcal{R}_3 \cup \mathcal{R}_4 \cup \mathcal{R}_5$, and then again with $i = 3$ for every restriction obtained with $i = 2$. This creates a set $\mathcal{R}'$ of $O(|V(G)|^9)$ restrictions. Finally, we construct $\mathcal{R}$ from $\mathcal{R}'$ by removing all restrictions that still contain lists which have size three for some vertex. Following Claims 3, 4, 5 and 6, the whole computation can be done in $O(|V(G)|^9 \cdot |V(G)|) = O(|V(G)|^{10})$ time.

Let us say that $x \in X$ is *wide* if $N_1(x) \neq \emptyset, N_2(x) \neq \emptyset$ and $N_3(x) \neq \emptyset$. Due to the construction of $\mathcal{R}'$ it holds that if $|L'(x)| = 3$ for some $(G', L', Z') \in \mathcal{R}'$, then x is wide.

It remains to show that (G, L, Z) is colorable if and only if $\mathcal{R}$ is colorable. By Claims 3, 4, 5 and 6, we know that if $\mathcal{R}$ is colorable then (G, L, Z) is colorable. Now assume that (G, L, Z) is colorable, and let c be a coloring of (G, L, Z). Consider any wide vertex x. Since the neighborhood of x can only have two distinct colors in total, there are two vertices $n_j \in N_j(x)$ and $n_k \in N_k(x)$ such that $c(n_j) = c(n_k) = i$, for some distinct $j, k \in \{1, 2, 3\}$. Then c is a type A, type B, or type C coloring with respect to i. Consequently, by Claims 3, 4, 5, there is a restriction $(G', L', Z') \in \mathcal{R}'$ that is colorable, and where for every wide vertex $y \in X \cap V(G')$ it holds that $|L'(y)| \leq 2$. Therefore

$(G', L', Z') \in \mathcal{R}$.

We now come to the running time analysis. Using Lemma 3.1.3, we can check in $O(|V(G)| + |E(G)|)$ time whether a given restriction (G', L', Z') of (G, L, Z) is colorable. Since we have $O(|V(G)|^9)$ many restrictions to consider, and these can be computed in $O(|V(G)|^{10})$ time, the total running time amounts to $O(|V(G)|^9(|V(G)| + |E(G)|))$. This completes the proof.

Chapter 4

Obstructions for three-coloring and list three-coloring H-free graphs

4.1 Introduction

A graph is 4-vertex-critical H-free if it has no induced subgraph isomorphic to H, not 3-colorable, but all its proper induced subgraphs are 3-colorable. In [10], we proved the following theorem, solving a problem posed by Golovach *et al.* [24] and answering a question of Seymour [53] (see Chapter 2 for formal definitions).

4.1.1 (Chudnovsky *et al.* [10])**.** *Let H be a connected graph. There are only finitely many 4-vertex-critical H-free graphs if and only if H is an induced subgraph of P_6.*

In view of our result, Golovach *et al.* [24] posed the problem of extending the above theorem to a complete dichotomy for arbitrary graphs H. While this seems to be an incremental question at first sight, it requires entirely different machinery to be settled.

A second natural generalization of Theorem 4.1.1 is to go from 4-vertex critical graphs to minimal obstructions to list 3-colorability. This more technical generalization is motivated, among other things, by a theorem of Jansen and Kratsch [33], that says that if there is a finite list of H-free minimal obstructions to list-3-colorability, then a polynomial kernelization of the 3-coloring problem exists when parameterized by the number of vertices needed to hit all induced copies of H. This application is outside of the scope of this thesis, and so we skip the precise definitions.

In this chapter, We answer both questions mentioned above. We obtain the following dichotomy theorem which now fully settles the problem of characterizing all graphs H for which there are only finitely many 4-vertex-critical H-free graphs.

4.1.2. *Let H be a graph. There are only finitely many H-free 4-vertex-critical graphs if and only if H is an induced subgraph of P_6, $2P_3$, or $P_4 + kP_1$ for some $k \in \mathbb{N}$.*

The tools used in [10] to prove Theorem 4.1.1 were tailored specifically for the P_6-free case and do not generalize well, while our new approach is significantly more powerful. The idea is to transfer the problem to the more general list setting and solve it there, showing that there is a constant C such that minimal obstructions have bounded size at most C. This generality comes at a certain cost: the upper bound we get is far from sharp.

Our second main result is the analogue of Theorem 4.1.2 in the list setting. Recall that we say a pair (G, L) with $L(v) \subseteq \{1, 2, 3\}$, $v \in V(G)$, a *minimal list-obstruction* if (G, L) is not colorable but for all proper induced subgraphs A of G the pair (A, L) is colorable. Here and throughout the thesis, if H is an induced subgraph of G, then by (H, L) we mean H with the list system L restricted to $V(H)$. We prove the following theorem.

4.1.3. *Let H be a graph. There are only finitely many H-free minimal list-obstructions if and only if H is an induced subgraph of P_6, or of $P_4 + kP_1$ for some $k \in \mathbb{N}$.*

Note that there are infinitely many $2P_3$-free minimal list-obstructions while there are only finitely many 4-vertex-critical $2P_3$-free graphs. Thus, Theorem 4.1.2 is not a special case of Theorem 4.1.3. Moreover, the fact that there are only finitely many P_6-free minimal list-obstructions does not follow from Theorem 4.1.1.

4.1.1 Structure of this Chapter

In Section 4.2 we develop the concept of the so-called *propagation path*, which is the main tool in showing that there are only finitely many H-free minimal list-obstructions (for the right choices of H) with lists of size at most 2. In particular, we show that for every minimal list-obstruction with lists of size at most 2, we can delete at most four vertices so that what remains is the union of four propagation paths.

In Section 4.3 we prove that there are only finitely many P_6-free minimal list-obstructions. We split the proof into two parts. In the first part we show that there are only finitely many P_6-free minimal list-obstructions where every list is of size at most 2, which amounts to studying P_6-free propagation paths. This step has a computer-aided proof. We also have a computer-free proof of this fact, but it is tedious, and we decided to only include a sketch of it. In the second part of the proof we reduce the general problem to the case solved in the first part. Here we rely on a structural analysis, making use of a structure theorem for P_t-free graphs.

Using a similar approach, in Section 4.4 we prove that there are only finitely many $2P_3$-free 4-vertex-critical graphs (of course, certain modifications are needed, because the list version is false in this case). In Section 4.5 we show that there are only finitely many $P_4 + kP_1$-free minimal list-obstructions.

In Section 4.6 we prove the necessity in the statement of Theorem 4.1.2 and Theorem 4.1.3, providing infinite families of H-free 4-vertex-critical graphs and minimal list-obstructions.

Our main results are proven in Section 4.7 where we put together the results mentioned above.

4.2 Obstructions with lists of size at most two

The aim of this section is to provide an upper bound on the order of the H-free minimal list-obstructions in which every list has at most two entries. Let us stress the fact that we restrict our attention to lists which are (proper) subsets of $\{1, 2, 3\}$. Before we state our lemma, we need to introduce the following technical definition.

Let (G, L) be a minimal list-obstruction such that $|L(v)| \leq 2$ for all $v \in V(G)$. Let $P = v_1$-v_2-$\ldots$-v_k be a path in G, not necessarily induced. Assume that $|L(v_1)| \geq 1$ and $|L(v_i)| = 2$ for all $i \in \{2, \ldots, k\}$. Moreover, assume that there is a color $\alpha \in L(v_1)$ such that if we give color α to v_1 and update along P, we obtain a coloring c of P. Please note that c may not be a coloring of the graph $G|V(P)$. For $i \in \{2, \ldots, k\}$ with $L(v_i) = \{\beta, \gamma\}$ and $c(v_i) = \beta$ we define the *shape* of v_i to be $\beta\gamma$, and denote it by $S(v_i)$. If every edge v_iv_j (of G) with $3 \leq i < j \leq k$ and $i \leq j - 2$ is such that

$$S(v_i) = \alpha\beta \text{ and } S(v_j) = \beta\gamma, \tag{4.1}$$

where $\{1, 2, 3\} = \{\alpha, \beta, \gamma\}$, then we call P a *propagation path* of G and say that P *starts with color* α. As we prove later, (4.1) implies that the updating process from v_1 along P to v_k cannot be

shortcut via any edge v_iv_j with $3 \leq i < j \leq k$ and $i \leq j-2$.

The next lemma shows that, when bounding the order of our list-obstructions, we may concentrate on upper bounds on the size of propagation paths.

4.2.1. *Let (G, L) be a minimal list-obstruction, where $|L(v)| \leq 2$ and $L(v) \in \{1,2,3\}$ for every $v \in V(G)$. Assume that all propagation paths in G have at most λ vertices for some $\lambda \geq 20$. Then $|V(G)| \leq 4\lambda + 4$.*

In the next section we prove the above lemma. First we show that if G is a minimal list-obstruction in which every list contains at most two colors, then $V(G) = V_1 \cup V_2$, where $|V_1 \cap V_2| \geq 1$, and for some $v \in V_1 \cap V_2$, each $G|V_i$ has a Hamiltonian path P_i starting at v. Moreover, if $L(v) = \{c_1, c_2\}$, then for every $i \in \{1,2\}$ giving v the color c_i and updating along P_i, results in a pair of adjacent vertices of G receiving the same list of size one. Then we prove that the edges of G that are not the edges of P_1, P_2 are significantly restricted, and consequently each P_i is (almost) the union of two propagation paths, thus proving the lemma.

4.2.1 Proof of Lemma 4.2.1

Let $(G = (V, E), L)$ be a minimal list-obstruction such that $|L(v)| \leq 2$ and $L(v) \subseteq \{1,2,3\}$ for all $v \in V$. If there is a vertex with an empty list, then this is the only vertex of G and we are done. So, we may assume that every vertex of G has a non-empty list. Let $V_1 = \{v \in V : |L(v)| = 1\}$ and $V_2 = \{v \in V : |L(v)| = 2\}$.

(1) *Let $x \in V$ and $\alpha \in L(x)$ be arbitrary. Assume we give color α to x and update exhaustively in the graph $(G|(V_2 \cup \{x\}), L)$. Let c be partial coloring thus obtained. For each $y \in V_1$ that did not receive a color so far, let $c(y)$ be the unique color in $L(y)$. Then there is an edge uv such that $c(u) = c(v)$.*

Proof. Let us give color α to x and update exhaustively, but only considering vertices and edges in the graph $(G|(V_2 \cup \{x\}), L)$. We denote this partial coloring by c. For each $y \in V_1$ that did not receive a color so far, let $c(y)$ be the unique color in $L(y)$. For a contradiction, suppose that this partial coloring c is proper.

Since G is an obstruction, c is not a coloring of G, meaning there are still vertices with two colors left on their list. We denote the set of these vertices by U. By minimality of G, we know

that both graphs $(G', L') := (G \setminus U, L)$ and $(G'', L'') := (G|(U \cup V_1) \setminus x, L)$ are colorable and have at least one vertex.

Let c' be the coloring of G' such that $c'(u) = c(u)$ for all $u \in V(G')$, and let c'' be a coloring of G''. It is clear that c' and c'' agree on the vertices in $V(G') \cap V(G'') = V_1 \setminus \{x\}$. Moreover, if $v \in (V_2 \setminus U) \cup \{x\}$ and $u \in U$ such that $uv \in E(G)$, then $c(v) \notin L(u)$. Since $c'(v) = c(v)$ for every $v \in V(G')$, we deduce that $c'(v) \neq c''(u)$ for every $uv \in E(G)$ with $u \in U$ and $v \in (V_2 \setminus U) \cup \{x\}$. Consequently, we found a coloring of (G, L), a contradiction. □

(2) *It holds that* $|V_1| \leq 2$.

Proof. Suppose that $|V_1| \geq 3$, and let $x \in V_1$ and $\alpha \in L(x)$. Let us give color α to x and update exhaustively, but only considering vertices and edges in the graph $(G|(V_2 \cup \{x\}), L)$. We denote this partial coloring by c.

Since G is minimal, there is no edge uv with $u, v \in V_2 \cup \{x\}$ and $c(u) = c(v)$. Since $(G|(V_1 \setminus \{x\}), L)$ is colorable by the minimality of G, and since $|L(v)| = 1$ for every $v \in V_1 \setminus \{x\}$, it follows that there is an edge uv with $u \in V_2 \cup \{x\}$ and $v \notin V_2 \cup \{x\}$ such that $L(v) = \{c(u)\}$. It follows that $(G|(V_2 \cup \{x, v\}), L)$ is not colorable, and so by the minimality of G, $V_1 = \{x, v\}$, as required. This proves the first claim. □

Next we prove that, loosely speaking, G is the union of at most two paths, starting at a common vertex, updating along each of which yields an improper partial coloring. Depending on the cardinality of V_1, we arrive at three different situations which are described by the following three claims.

(3) *Assume that* $|V_1| = 0$, *and pick* $x \in V$ *arbitrarily. Let us say that* $L(x) = \{1, 2\}$. *For* $\alpha = 1, 2$ *there is a path* $P^\alpha = v_1^\alpha$-$\ldots$-$v_{k_\alpha}^\alpha$, *not necessarily induced, with the following properties.*

(a) If we give color α *to* x *and update along* P^α, *then all vertices of* P^α *will be colored.*

(b) Assume that v_i^α *gets colored in color* γ_i, $i = 1, \ldots, k_\alpha$. *Then there is an edge of the form* $v_i^\alpha v_j^\alpha$ *with* $\gamma_i = \gamma_j$.

(c) $V = V(P^1) \cup V(P^2)$.

Proof. We give color α to x and update exhaustively from x. According to Claim 1, after some round of updating an edge appears whose end vertices receive the same color. We then stop the updating procedure. During the whole updating procedure we record an auxiliary digraph $D = (W, A)$ as follows. Initially, $W = \{x\}$ and $A = \emptyset$. Whenever we update a vertex u from a vertex v, we add the vertex u to W and the edge (v, u) to A. This gives a directed tree whose root is x.

We can find directed paths R and S in T both starting in x and ending in vertices y and z, say, such that y and z are adjacent in G and they receive the same color during the updating procedure. We may assume that $R = u_1\text{-}\dots\text{-}u_k\text{-}v_1\text{-}\dots\text{-}v_r$ and $S = u_1\text{-}\dots\text{-}u_k\text{-}w_1\text{-}\dots\text{-}w_s$, where R and S share only the vertices $u_1, \dots, u_k$. For each vertex $v \in V(R) \cup V(S)$, let $c(v)$ be the color received by v in the updating procedure. Moreover, let $c'(v)$ be the unique color in $L(v) \setminus \{c(v)\}$. Observe that, setting $w_0 = v_0 = u_k$, we have that $c'(w_i) = c(w_{i-1})$ for every $i \in \{1, \dots, s\}$, and $c'(v_i) = c(v_{i-1})$ for every $i \in \{1, \dots, k\}$.

Consider the following, different updating with respect to x. We again give color α to x, and then update along R. Now we update w_s from v_r, thus giving it color $c'(w_s)$. This, in turn, means we can update w_{s-1} from w_s, giving it color $c'(w_{s-1})$, and so on. Finally, when we update w_1 and it receives color $c'(w_1)$, an edge appears whose end vertices are colored in the same color. Indeed, $u_k w_1$ is such an edge since $c(u_k) = c'(w_1)$. Summing up, the path

$$P^\alpha = u_1\text{-}\dots\text{-}u_k\text{-}v_1\text{-}\dots\text{-}v_r\text{-}w_s\text{-}w_{s-1}\text{-}\dots\text{-}w_1$$

starts in x and, when we give x the color α and update along P^α, we obtain an improper partial coloring. As $\alpha \in \{1, 2\}$ was arbitrary, the assertions (a) and (b) follow.

To see (c), just note that the graph $G|(V(P_1) \cup V(P_2))$ is an obstruction: giving either color of $L(x)$ to x and updating exhaustively yields a monochromatic edge. By the minimality of G, $G = G|(V(P_1) \cup V(P_2))$ and so (c) holds. □

(4) *Assume that $|V_1| = 1$, say $V_1 = \{x\}$ with $L(x) = \{\alpha\}$. Then the following holds:*

(a) there is a Hamiltonian path $P = v_1\text{-}\dots\text{-}v_k$ of G with $x = v_1$;

(b) updating from $x = v_1$ along P assigns a color γ_i to v_i, $i = 1, \dots, k$; and

(c) there is an edge of the form $v_i v_j$ with $\gamma_i = \gamma_j$.

Proof. We assign color α to x and update exhaustively from x. Let c be the obtained partial coloring. According to Claim 1, there is an edge uv of G with $c(u) = c(v)$. Since (G, L) is a minimal obstruction, every vertex of G received a color in the updating process: otherwise, we could simply remove such a vertex and still have an obstruction.

Repeating the argument from the proof of Claim 3, we obtain a path P that starts in x and, when we give x color α and update along P, we obtain an improper partial coloring. This proves (b) and (c). Due to the minimality of (G, L), P is a Hamiltonian path, which proves (a). □

(5) *Assume* $|V_1| = 2$, *say* $V_1 = \{x, y\}$ *with* $L(x) = \{\alpha\}$ *and* $L(y) = \{\beta\}$. *Then the following holds:*

(a) there is a Hamiltonian path $P = v_1$-$\ldots$-v_k *of* G *with* $x = v_1$ *and* $y = v_k$; *and*

(b) updating from v_1 *along* P *assigns the color* β *to* v_{k-1}.

Proof. We color x with color α and update exhaustively from x, but only considering vertices and edges of the graph $G \setminus y$. Let c be the obtained partial coloring. By minimality, c is proper. According to Claim 1, there is a neighbor u of y in G with $c(u) = \beta$.

Like in the proof of Claim 3 and Claim 4, we see that there is a path P from x to y whose last edge is uy such that giving color α to x and then updating along P implies that u is colored with color β, which implies (b). Due to the minimality of (G, L), P is a Hamiltonian path , and thus (a) holds. □

We can now prove our main lemma.

Proof of Lemma 4.2.1. Recall from Claim 2 that $|V_1| \leq 2$.

Case $|V_1| = 0$**.** For this case Claim 3 applies and we obtain x, P^1 and P^2 as in the statement of the claim. We may assume that, among all possible choices of x, P^1 and P^2, the value $\max\{|V(P^1)|, |V(P^2)|\}$ is minimum and, subject to this, $\min\{|V(P^1)|, |V(P^2)|\}$ is minimum.

Let us say that $P^1 = v_1$-v_2-$\ldots$-v_s, where $v_1 = x$. Consider v_1 to be colored in color 1, and update along P^1, but only up to v_{s-1}. Due to the choice of P^1 and P^2 being of minimum length, the coloring so far is proper. Now when we update from v_{s-1} to v_s, two adjacent vertices receive the same color. Let the partial coloring obtained so far be denoted c. Let X be the set of neighbors w of v_s in $V(P^1)$ with $c(w) = c(v_s)$, and let r be minimum such that $v_r \in X$.

We claim that $s - r \leq \lambda$. To see this, let $c'(v_j)$ be the unique color in $L(v_j) \setminus \{c(v_j)\}$, for all $j = 1, \ldots, s$. We claim that the following assertions hold.

(a) $c(v_j) = c'(v_{j+1})$ for all $j = r, \ldots, s-1$.

(b) For every edge v_iv_j with $r \leq i, j \leq s-1$ it holds that $c(v_i) \neq c(v_j)$.

(c) For every edge v_iv_j with $r \leq i, j \leq s$ and $j - i \geq 2$ it holds that $c(v_i) \neq c'(v_j)$.

(d) For every edge v_iv_j with $r+2 \leq i, j \leq s$ it holds that $c'(v_i) \neq c'(v_j)$.

Assertion (a) follows from the fact that P obeys the assertions of Claim 3. For (b), note that the choice of P_1 to be of minimum length implies that until we updated v_s, the partial coloring is proper.

To see (c), suppose there is an edge v_iv_j with $r \leq i, j \leq s$ and $j - i \geq 2$ such that $c(v_i) = c'(v_j)$. Then the path P^1 can be shortened to the path v_1-$\ldots$-v_i-v_j-$\ldots$-v_s, which is a contradiction.

Now we turn to (d), and consider the following coloring. We color P^1 as before up to v_r. Now we update from v_r to v_s, giving color $c'(v_s)$ to v_s. Then we color v_{s-1} with color $c'(v_{s-1})$, then v_{s-2} with color $c'(v_{s-2})$, and so on, until we reach v_{r+1}. But $c'(v_{r+1}) = c(v_r)$ due to (a), which means that the path $Q^1 = v_1$-v_2-$\ldots$-v_r-v_s-v_{s-1}-v_{s-2}-$\ldots$-v_{r+1} is a choice equivalent to P^1. In particular, due to the choice of P^1 and P^2, the constructed coloring of Q^1 is proper if we leave out v_{r+1}. Hence, there is no edge v_iv_j with $r+2 \leq i, j \leq s$ such that $c'(v_i) = c'(v_j)$. This yields (d).

From (a)-(d) it follows that every edge v_iv_j with $r+2 \leq i < j \leq s$ and $i \leq j-2$ is such that

$$S(v_i) = \alpha\beta \text{ and } S(v_j) = \beta\gamma, \tag{4.2}$$

where $\{1,2,3\} = \{\alpha, \beta, \gamma\}$. Consequently, the path v_r-$\ldots$-v_{s-1} is a propagation path. By assumption, $|\{v_r, \ldots, v_{s-1}\}| \leq \lambda$ and so $s - r \leq \lambda$.

A symmetric consideration holds for P^2. Let us now assume that $|V(P^1)| \geq |V(P^2)|$. It remains to show that r is bounded by some constant. To this end, recall that $\lambda \geq 20$.

Suppose that there is an edge v_iv_j with $3 \leq i \leq j \leq r$ such that $c'(v_i) = c'(v_j)$. We then put $x' = v_j$, $Q^1 = v_j$-$\ldots$-v_s, and $Q^2 = v_j$-$\ldots$-v_i. But this is a contradiction to the choice of x, P^1, and P^2, as $\max\{|V(Q^1)|, |V(Q^2)|\} < \max\{|V(P^1)|, |V(P^2)|\}$. In addition to the assertion we just proved, which corresponds to assertion (d) above, the assumptions (a)-(c) from above also hold here, where we replace r by 1 and s by r. Hence, using the same argumentation as above, we see that $r \leq \lambda + 1$. Summing up, we have $|V| \leq |V(P^1) \cup V(P^2)| \leq 2|V(P^1)| \leq 4\lambda + 2$, as desired.

Case $|V_1| = 1$. Now Claim 4 applies and we obtain the promised path, say $P = v_1\text{-}\ldots\text{-}v_s$, with $|L(v_1)| = 1$. Let us say $L(v_1) = \{1\}$. Consider v_1 to be colored in color 1, and update along P, but only up to v_{s-1}. Due to the choice of P, the coloring so far is proper. Now when we update from v_{s-1} to v_s, two adjacent vertices receive the same color. Let the partial coloring obtained so far be denoted c. Let X be the set of neighbors w of v_s on P with $c(w) = c(v_s)$, and let r be minimum such that $v_r \in X$. Moreover, let $c'(v_j)$ be the unique color in $L(v_j) \setminus \{c(v_j)\}$, for all $j = 2, \ldots, s$. Just like in the case $|V_1| = 0$, we obtain the assertions (a)-(d) from above and this implies $s - r \leq \lambda$.

It remains to show that $r \leq \lambda + 1$. To see this, suppose that there is an edge v_iv_j with $2 \leq i \leq j \leq r$ such that $c'(v_i) = c'(v_j)$. We then put $P^1 = v_j\text{-}\ldots\text{-}v_s$ and $P^2 = v_j\text{-}\ldots\text{-}v_i$. Now, if we give color $c(v_j)$ to v_j and update along P^1 we obtain an improper coloring. Moreover, if we give color $c'(v_j)$ to v_j and update along P^2 we also obtain an improper coloring. This means that the pair $(G|(V(P^1) \cup V(P^2)), L)$ is not colorable, in contradiction to the minimality of (G, L).

The assertion we just proved corresponds to assertion (d) above, and the assumptions (a)-(c) also hold here, where we replace r by 1 and s by r. Hence, we know $r \leq \lambda + 1$ and obtain $|V| = |V(P)| \leq 2\lambda + 1$.

Case $|V_1| = 2$. Claim 5 applies and we obtain the promised path, say $P = v_1\text{-}\ldots\text{-}v_s$, with $|L(v_1)| = |L(v_s)| = 1$. Let us say $L(v_1) = \{\alpha\}$ and $L(v_s) = \{\beta\}$. Consider v_1 to be colored in color α, and update along P, but only up to v_{s-1}. Due to the choice of P, the partial coloring so far is proper. Let the partial coloring obtained so far be denoted c, and put $c(v_s) = \beta$. We now have $c(v_{s-1}) = c(v_s)$, and this is the unique pair of adjacent vertices of G that receive the same color.

For each $j = 2, \ldots, s-1$, we denote by $c'(v_j)$ the unique color in $L(v_j) \setminus \{c(v_j)\}$. We will show that $s \leq \lambda + 1$. Just like in the cases above the following assertions apply.

(a) $c(v_j) = c'(v_{j+1})$ for all $j = 1, \ldots, s-2$.

(b) For every edge v_iv_j with $1 \leq i, j \leq s-1$ it holds that $c(v_i) \neq c(v_j)$.

(c) For every edge v_iv_j with $1 \leq i, j \leq s-1$ and $j - i \geq 2$ it holds that $c(v_i) \neq c'(v_j)$.

(d) For every edge v_iv_j with $3 \leq i, j \leq s-1$ it holds that $c'(v_i) \neq c'(v_j)$.

Let $r' = 1$ and $s' = s-1$. As above we see that $s' - r' \leq \lambda - 1$. Hence, $s \leq \lambda + 1$. From the fact that

P is a Hamiltonian path in G we obtain the desired bound $|V| = |V(P)| \leq \lambda + 1$. This completes the proof. □

4.3 P_6-free minimal list-obstructions

The aim of this section is to prove that there are only finitely many P_6-free minimal list-obstructions. In Section 4.3.1 we prove the following lemma which says that there are only finitely many P_6-free minimal list-obstructions with lists of size at most two.

4.3.1. *Let (G, L) be a P_6-free minimal list-obstruction for which $|L(v)| \leq 2$ and $L(v) \subseteq \{1, 2, 3\}$ holds for all $v \in V(G)$. Then $|V(G)| \leq 100$.*

Our proof of this lemma is computer-aided. We also have a computer-free proof, but it is tedious and complicated, and gives a significantly worse bound on the size of the obstructions, so we will only sketch the idea of the computer-free proof.

In Section 4.3.3 we solve the general case, where each list may have up to three entries, making extensive use of Lemma 4.3.1. We prove the following lemma.

4.3.2. *There exists an integer C such that the following holds. Let G be a P_6-free graph, and let L be a list system such that $L(v) \subseteq \{1, 2, 3\}$ for every $v \in V(G)$. Suppose that (G, L) is a minimal list obstruction. Then $|V(G)| \leq C$. Consequently, there are only finitely many P_6-free minimal list-obstructions.*

The main technique used in the proof of Lemma 4.3.2 is to guess the coloring on a small set S of vertices of the minimal list-obstruction at hand, (G, L) say. After several transformations, we arrive at a list-obstruction (G, L') where each list has size at most two, and so we may apply Lemma 4.3.1 to show that there is a minimal list-obstruction (H, L') with a bounded number of vertices induced by (G, L'). We can prove that G is essentially the union of these graphs H (one for each coloring of S), and so the number of vertices of G is bounded by a function of the number of guesses we took in the beginning. Since we precolor only a (carefully chosen) small part of the graph, we can derive that the number of vertices of G is bounded by a constant.

To find the right vertex set to guess colors for, we use a structure theorem for P_t-free graphs [8] that implies the existence of a well-structured connected dominating subgraph of a minimal list-obstruction.

4.3.1 Proof of Lemma 4.3.1

Let (G, L) be a P_6-free minimal list-obstruction such that every list contains at most two colors. Suppose that $P = v_1\text{-}\ldots\text{-}v_k$ is a propagation path in (G, L). We show that if G is P_6-free, then $k \leq 24$. In view of Lemma 4.2.1, this proves that G has at most 100 vertices.

Our proof is computer-aided, but conceptually very simple. The program generates the paths v_1, v_1-v_2, v_1-v_2-v_3, and so on, lists for each v_i, as in the definition of a propagation path, and edges among the vertices in the path. Whenever a P_6 or an edge violating condition (4.1) of the definition of a propagation path is found, the respective branch of the search tree is closed. Since the program does not find such a path on 25 vertices (cf. Table 1), our claim is proved.

Vertices	1	2	3	4	5	6	7	8	
Propagation paths	1	2	6	22	86	350	1 220	2 656	
Vertices	9	10	11	12	13	14	15	16	
Propagation paths	4 208	5 360	5 864	5 604	5 686	5 004	4 120	3 400	
Vertices	17	18	19	20	21	22	23	24	25
Propagation paths	2 454	1 688	1 064	516	202	72	18	2	0

Table 4.1: Counts of all P_6-free propagation paths with lists of size 2 meeting condition (4.1) generated by Algorithm 1.

The pseudocode of the algorithm is shown in Algorithm 1 and 2. Our implementation of this algorithm can be downloaded from [21]. Table 4.1 lists the number of configurations generated by our program.

Algorithm 1 Generate propagation paths and lists

1: $H = (\{v_2\}, \emptyset)$
2: $c(v_1) = 1$
3: $L(v_1) = \{1\}$
4: Construct(H, c, L)
// We may assume $c(v_1) = 1$ and $L(v_1) = \{1\}$.

Next we sketch the idea of the computer-free proof. Let (G, L) be a minimal list obstruction with

Algorithm 2 Construct(Graph H, coloring c, list system L)

1: $j = |V(H)|$
2: $V(H) = V(H) \cup \{v_{j+1}\}$
3: $E(H) = E(H) \cup \{v_j v_{j+1}\}$
// This extends the path by the next vertex v_{j+1}.
4: **for all** $\alpha \in \{1,2,3\} \setminus \{c(v_j)\}$ and all $I \subseteq \{1,2,\ldots,j-1\}$ **do**
5: $H' = H$
6: $E(H') = E(H') \cup \{v_i v_{j+1} : i \in I\}$
// This adds edges from v_{j+1} to earlier vertices in all possible ways.
7: $c(v_{j+1}) = \alpha$
8: $L(v_{j+1}) = \{\alpha, c(v_j)\}$
9: **if** (H', c, L) is P_6-free and satisfies condition (4.1) **then**
10: Construct(H', c, L)
// If the propagation path is not pruned, we extend it further.
11: **end if**
12: **end for**

all lists of size at most two, and suppose for a contradiction that there is a (very) long propagation path P in G. We may assume that G does not contain a clique with four vertices. It follows from the main result of [20] that $G|V(P)$ contains a large induced subgraph H, which is a complete bipartite graph; let (A, B) be a bipartition of H. Using Ramsey's Theorem [48] we may assume that all vertices of A have the same shape, and all vertices of B have the same shape (by "coloring" the edges of H by the shapes of their ends). We can now choose a large subset A' of A all of whose members are pairwise far apart in P, and such that A' is far in P from some subset B' of B. We analyze the structure of short subpaths of P containing each $a \in A$, and the edges between such subpaths, and to B'. We can again use Ramsey's Theorem to assume that the structure is the same for every member of A' and every member of B'. Finally, we accumulate enough structural knowledge to find a P_6 in G, thus reaching a contradiction. If instead of P_6 we wanted to use the same method to produce P_5, the argument becomes much shorter, and it was carried out in [60].

4.3.2 Reducing obstructions

In this section we prove three lemmas which help us reduce the size of the obstructions. These lemmas will be used in the proofs of Sections 4.3.3 and 4.4.2.

Let (G, L) be a list-obstruction and let R be an induced subgraph of G. Let $\mathcal{L}$ be a set of subsystems of L satisfying the following assertions.

1. For every $L' \in \mathcal{L}$ there exists an induced subgraph $R(L')$ of R such that $|L'(v)| = 1$ for every $v \in V(R(L'))$.
2. For each $L' \in \mathcal{L}$, $L'(v) = L(v)$ for $v \in V(G) \setminus R(L')$ and $L'(v) \subseteq L(v)$ for $v \in R(L')$.
3. For every L-coloring c of R there exists a list system $L' \in \mathcal{L}$ such that $c(v) \in L'(v)$ for every $v \in R(L')$.

We call $\mathcal{L}$ a *refinement of L with respect to R*. Observe that $\{L\}$ with $R(L)$ being the empty graph is a refinement of L with respect to G, though this is not a useful refinement. For each list system $L' \in \mathcal{L}$ it is clear that (G, L') is again a list-obstruction, though not necessarily a minimal one, even if (G, L) is minimal.

4.3.3. *Assume that (G, L) is a minimal list-obstruction. Let R be an induced subgraph of G, and let $\mathcal{L} = \{L_1, L_2, \ldots, L_m\}$ be a refinement of L with respect to R. For every $L_i \in \mathcal{L}$, let (G_{L_i}, L_i) be a minimal obstruction induced by (G, L_i).*

Then $V(G) = R \cup \bigcup_{L_i \in \mathcal{L}} V(G_{L_i})$. Moreover, if each G_{L_i} can be chosen such that $|V(G_{L_i} \setminus R| \leq k$, then G has at most $|V(R)| + km$ vertices.

Proof. Let $(G_i, L_i|_{G_i})$ be a minimal list-obstruction induced by (G, L_i) such that $|V(G_i) \setminus V(R)| \leq k$, $i = 1, \ldots, m$. Suppose for a contradiction that there exists a vertex v in $V(G) \setminus R$ such that v is contained in none of the G_i, $i = 1, \ldots, m$. By the minimality of (G, L), $G \setminus \{v\}$ is L-colorable. Let c be an L-coloring of $G \setminus \{v\}$. We may assume that $c(r) \in L_1(r)$ for every $r \in V(R(L_1)) \cap V(G_1)$. Then c is a coloring of (G_1, L_1), which is a contradiction. This proves the first assertion of Lemma 4.3.3. Consequently,

$$|V(G)| \leq |\bigcup_{i=1}^{m} V(G_i)| \leq |V(R)| + |\bigcup_{i=1}^{m} V(G_i \setminus R)|$$

and the second assertion follows.

□

Next we prove a lemma which allows us to update three times with respect to a set of vertices with lists of size 1.

4.3.4. *Let (G, L) be a list-obstruction, and let $X \subseteq V(G)$ be a vertex subset such that $|L(x)| = 1$ for every $x \in X$. Let L' be the list obtained by updating with respect to X three times. Let (G', L') be a minimal list-obstruction induced by (G, L'). Then there exists a minimal list-obstruction induced by (G, L), say (G'', L), with $|V(G'')| \leq 36|V(G')|$.*

Proof. Let $Y = X_1$ and $Z = X_2$, as in the definition of updating i times. We choose sets R, S, and T as follows.

- For every $v \in V(G') \setminus (X \cup Y \cup Z)$, define $R(v)$ to be a minimum subset of $(X \cup Y \cup Z) \cap N(v)$ such that $\bigcup_{s \in R(v)} L'(s) = L(v) \setminus L'(v)$, and let $R = \bigcup_{v \in V(G') \setminus (X \cup Y \cup Z)} R(v)$.
- For every $v \in (V(G') \cup R) \cap Z$, define $S(v)$ to be a minimum subset of $(X \cup Y) \cap N(v)$ such that $\bigcup_{s \in S(v)} L'(s) = L(v) \setminus L'(v)$, and let $S = \bigcup_{v \in (V(G') \cup R) \cap Z} S(v)$.
- For every $v \in (V(G') \cup R \cup S) \cap Y$, define $T(v)$ to be a minimum subset of $X \cap N(v)$ such that $\bigcup_{s \in T(v)} L'(s) = L(v) \setminus L'(v)$, and let $T = \bigcup_{v \in (V(G') \cup R \cup S) \cap Y} T(v)$.

Clearly, $|R(v)| \leq 3$ for every $v \in V(G') \setminus (X \cup Y \cup Z)$, $|S(v)| \leq 2$ for every $v \in (V(G') \cup R) \cap Z$, and $|T(v)| \leq 2$ for every $v \in (V(G') \cup R \cup S) \cap Y$. Let $P = R \cup S \cup T \cup V(G')$, and observe that $|P| \leq (1+3+8+24)|V(G')| = 36|V(G')|$. It remains to prove that $(G|P, L)$ is not colorable. Suppose there exists a coloring c of $(G|P, L)$. Then c is not a coloring of (G', L'), and since $V(G') \subseteq P$, it follows that there exists $w \in V(G')$ such that $c(w) \notin L'(w)$. Therefore $c(w) \in L(w) \setminus L'(w)$.

We discuss the case when $v \in V(G') \setminus (X \cup Y \cup Z)$, as the cases of $v \in (V(G') \cup R) \cap Z$ and $v \in (V(G') \cup R \cup S) \cap Y$ are similar. We can choose $m \in R(w)$ such that $L'(m) = \{c(w)\}$ and one of the following holds.

- $m \in X$, and thus $L(m) = L'(m) = \{c(w)\}$.
- $m \in Y$, and thus for any $i \in L(m) \setminus L'(m)$ there exists $n_i \in T(m)$ such that $L(n_i) = \{i\}$.
- $m \in Z$, and thus for any $i \in L(m) \setminus L'(m)$ there exists $n_i \in S(m)$ such that either $L(n_i) = \{i\}$ or, for any $j \in L(n_i) \setminus \{i\}$, there exists $l_j \in T(n_i)$ with $L(l_j) = \{j\}$.

In all cases it follows that $c(m) = c(w)$, in contradiction to the fact that m and w are adjacent. This completes the proof. □

Let A be a subset of $V(G)$ and L be a list system; let c be an L-coloring of $G|A$, and let L_c be the list system obtained by setting $L_c(v) = \{c(v)\}$ for every $v \in A$ and updating with respect to A three times; we say that L_c is *obtained from L by precoloring A (with c) and updating three times*. If for every c, $|L_c(v)| \leq 2$ for every $v \in V(G)$, we call A a *semi-dominating set* of (G, L). If $L(v) = \{1, 2, 3\}$ for every $v \in G$ and A is a semi-dominating set of (G, L), we say that A is a *semi-dominating set* of G. Note that a dominating set is always a semi-dominating set. Last we prove a lemma for the case when G has a bounded size semi-dominating set.

4.3.5. *Let (G, L) be a minimal list-obstruction, and assume that G has a semi-dominating set A with $|A| \leq t$. Assume also that if (G', L') is a minimal obstruction where G' is an induced subgraph of G, and L' is a subsystem of L with $|L'(v)| \leq 2$ for every v, then $|V(G')| \leq m$. Then $|V(G)| \leq 36 \cdot 3^t \cdot m + t$.*

Proof. Consider all possible L-colorings $c_1, \ldots, c_s$ of A; then $s \leq 3^t$. For each i, let L_i be the list system obtained by updating with respect to A three times. Then $|L_i(v)| \leq 2$ for every $v \in V(G)$ and for every $i \in \{1, \ldots, s\}$. Now Lemma 4.3.3 together with Lemma 4.3.4 imply that $|V(G)| \leq 36 \cdot 3^t \cdot m + t$. This completes the proof. □

4.3.3 Proof of Lemma 4.3.2

We start with several claims that deal with vertices that have a special structure in their neighborhood.

(1) *Let G be a graph, and let $X \subseteq V(G)$ be connected. If $v \in V(G) \setminus X$ is mixed on X, then there exist adjacent $x_1, x_2 \in X$ such that v is adjacent to x_1 and non-adjacent to x_2.*

Proof. Since v is mixed on X, both the sets $N(v) \cap X$ and $X \setminus N(v)$ are non-empty, and since X is connected, there exist $x_1 \in N(v) \cap X$ and $x_2 \in X \setminus N(v)$ such that x_1 is adjacent to x_2. This proves Claim 1. □

(2) *Let G be a P_6-free graph and let $v \in V(G)$. Suppose that $G|N(v)$ is a connected bipartite graph with bipartition (A, B). Let G' be obtained from $G \setminus (A \cup B)$ by adding two new vertices a, b with*

$N_{G'}(a) = \{b\} \cup \bigcup_{u \in A}(N_G(u) \cap V(G'))$ *and* $N_{G'}(b) = \{a\} \cup \bigcup_{u \in B}(N_G(u) \cap V(G'))$. *Then* G' *is* P_6*-free.*

Proof. Suppose Q is a P_6 in G. Then $V(Q) \cap \{a, b\} \neq \emptyset$. Observe that if both a and b are in $V(Q)$, then $v \notin V(Q)$. If only one vertex of Q, say q, has a neighbor in $\{a, b\}$, say a, then we get a P_6 in G by replacing a with a neighbor of q in A, and, if $b \in V(Q)$, replacing b with v. Thus we may assume that two vertices q, q' of Q have a neighbor in $\{a, b\}$. If q and q' have a common neighbor $u \in A \cup B$, then $G|((V(Q) \setminus \{a, b\} \cup \{u\}))$ is a P_6 in G, a contradiction. So no such u exists, and in particular $v \notin V(Q)$. Let Q' be an induced path from q to q' with $V(Q') \setminus \{q, q'\} \subseteq A \cup B \cup \{v\}$, meeting only one of the sets A, B if possible. Then $G|((V(Q) \setminus \{a, b\}) \cup V(Q'))$ is a P_6 in G, a contradiction. This proves Claim 2. □

In the remainder of this section G is a P_6-free graph.

(3) *Let* (G, L) *be a minimal list-obstruction. Let* A *be a stable set in* G*. Let* U *be the set of vertices of* $V(G) \setminus A$ *that are not mixed on* A*, and let* $k = |V(G) \setminus (A \cup U)|$*. Then* $|A| \leq 7 \times 2^k$.

Proof. Partition A by the adjacency in $V(G) \setminus (A \cup U)$ and by lists; more precisely let $A = A_1 \cup A_2 \cup \ldots \cup A_{7 \times 2^k}$ such that for every i and for every $x, y \in A_i$, $N(x) \setminus (A \cup U) = N(y) \setminus (A \cup U)$ and $L(x) = L(y)$. Since A is stable and no vertex of U is mixed on A, it follows that for every $x, y \in A_i$, $N(x) = N(y)$. We claim that $A_i \leq 1$ for every i. Suppose for a contradiction that there exist $x, y \in A_i$ with $x \neq y$. By the minimality of (G, L), $G \setminus x$ is L-colorable. Since $N(x) = N(y)$ and $L(x) = L(y)$, we deduce that G is L-colorable by giving x the same color as y, a contradiction. This proves Claim 3. □

(4) *Let* (G, L) *be a minimal list-obstruction. Let* H *be an induced subgraph of* G *such that* H *is connected and bipartite. Let* (A, B) *be the bipartition of* H*, and let* $u \in V(G)$ *be complete to* $A \cup B$*. Let* U *be the set of all vertices in* $V(H) \setminus (A \cup B)$ *that are not mixed on* A*. Let* $K = V(G) \setminus (A \cup B \cup U)$ *and* $k = |K|$*. Then* $|A| \leq 7 \cdot 2^{7 \cdot 2^k + k}$.

Proof. We partition A according to the adjacency in K and the lists of the vertices. More precisely, let $A = A_1 \cup A_2 \cup \ldots \cup A_{7 \cdot 2^k}$ such that for any i and for any $x, y \in A_i$, $N(x) \cap K = N(y) \cap K$ and $L(x) = L(y)$. Analogously, let $B = B_1 \cup B_2 \cup \ldots \cup B_{7 \cdot 2^k}$.

Next, for $i = 1, \ldots, 7 \cdot 2^k$, we partition $A_i = A_i^1 \cup \ldots \cup A_i^{7 \cdot 2^k}$ according to the sets $B_1, B_{7 \cdot 2^k}$ in which they have neighbors. More precisely, for any t and any $x, y \in A_i^t$, $N(x) \cap B_j \neq \emptyset$ if and only if $N(y) \cap B_j \neq \emptyset$, $j = 1, \ldots, 7 \cdot 2^k$. We partition the sets in B analogously. We claim that $A_i^j \leq 1$. Suppose that there exist $x, y \in A_i^j$ with $x \neq y$. Let $N = N(x) \cap B$. Let C be the component of $H \setminus x$ with $y \in C$. Let N_1 be the set of vertices of N whose unique neighbor in H is x, and let $N_2 = N \setminus N_1$.

Suppose first that there is a vertex $s \in N_2 \setminus C$. Since y is not dominated by x, there exists $t \in (B \cap N(y)) \setminus N(x)$. Since $s \notin C$, it follows that s and t have no common neighbor in H, and so, since G is P_6-free, there is a 5-vertex path Q in H with ends s and t; let the vertices of Q be q_1-$\ldots$-q_5, where $s = q_1$ and $t = q_5$. Since $s \notin C$, it follows that $q_2 = x$. Since y-t-q_4-q_3-x-s is not a P_6 in G, it follows that y is adjacent to q_3, and so we may assume that $q_4 = y$. Let $p \in A \setminus \{x\}$ be a neighbor of s. Since p-s-x-q_3-y-t is not a P_6, it follows that p has a neighbor in $\{t, q_3\}$, contrary to the fact that $s \notin C$. This proves that $N_2 \subseteq C$.

Observe that $C \cap N_1 = \emptyset$. By the minimality of (G, L), there is a coloring c of $(G \setminus (N_1 \cup \{x\}), L)$. Since u is complete to $V(C)$, it follows that $C \cap A$ and $C \cap B$ are both monochromatic. We now describe a coloring of G. Color x with $c(y)$. Let $n_1 \in N_1$ be arbitrary. Since $x, y \in A_i^j$, there exists n_1' in B such that n_1' is adjacent to y, $L(n_1) = L(n_1')$, and n_1, n_1' have the same neighbors in K. Now color n_1 with $c(n_1')$. Repeating this for every vertex of N_1 produces a coloring of (G, L) a contradiction. This proves Claim 4. □

(5) *There is a function $q : \mathbb{N} \to \mathbb{N}$ such that the following holds. Let (G, L) be a minimal list-obstruction. Let $D_1, \ldots, D_t$ be connected subsets of $V(G)$ with the following properties.*

- *$|D_i| > 1$ for every i,*
- *$D_1, \ldots, D_t$ are pairwise disjoint and anticomplete to each other,*
- *for each i there is a set $U_i \subseteq V(G)$ such that U_i is complete to D_i,*
- *D_i is anticomplete to $V(G) \setminus (U_i \cup D_i)$,*
- *for every $i \in \{1, \ldots, t\}$ there is $c_i \in \{1, 2, 3\}$ such that $c(u) \neq c_i$ for every coloring c of (G, L) and for every $u \in U_i$, and*

- $V(G) \neq D_1 \cup U_1$.

Then there is an induced subgraph F of G such that $V(G) \setminus \bigcup_{i=1}^{t} D_i \subseteq V(F) \subseteq V(G)$ and a list system L' such that

- $|L'(v)| \leq 2$ *for every* $v \in V(F) \cap \bigcup_{i=1}^{t} D_i$,
- $L'(v) = L(v)$ *for every* $v \in V(F) \setminus \bigcup_{i=1}^{t} D_i$, *and*
- (F, L') *is a minimal list-obstruction, and* $|V(G)| \leq q(|V(F)|)$.

Proof. Write $D = \bigcup_{i=1}^{t} D_i$. Since $V(G) \neq U_1 \cup D_1$, it follows from the minimality of (G, L) that $(G|(D_i \cup U_i), L)$ is colorable for every i, and so each $G|D_i$ is bipartite. Let D_i^1, D_i^2 be the bipartition of $G|D_i$. Then for every i there is a coloring of $(G|D_i, L)$ in which each of the sets D_i^1, D_i^2 is monochromatic, and in particular $\bigcap_{d \in D_i^j} L(d) \neq \emptyset$ for every $i \leq t$ and $j \in \{1, 2\}$.

For every $i \leq t$ and $j \in \{1, 2\}$, let $d_i^j \in D_i^j$ such that d_i^1 is adjacent to d_i^2. Set $L''(d_i^j) = \bigcap_{d \in D_i^j} L(d)$. Let F'' be the graph obtained from G by deleting $D \setminus (\bigcup_{i=1}^{m} \{d_i^1, d_i^2\})$. Set $L''(v) = L(v)$ for every $v \in V(F'') \setminus D$.

We may assume that there exists $s \in \{0, 1, \ldots, t\}$ such that $|L''(d_i^1)| = 3$ for every $i \leq s$, and that for $i \in \{s+1, \ldots, t\}$ and $j \in \{1, 2\}$, $|L''(d_i^j)| \leq 2$.

Let F' be obtained from F'' by deleting $\{d_1^1, \ldots, d_s^1\}$. For $i \in \{1, \ldots, s\}$, let $L'(d_i^2) = L''(d_i^1) \setminus \{c_i\}$. Let $L'(v) = L''(v)$ for every other vertex of F'. Then $V(G) \setminus D \subseteq V(F')$, $L'(v) = L(v)$ for every $v \in V(F') \setminus D$, and $|L'(v)| \leq 2$ for $v \in V(F') \cap D$.

We claim that (F', L') is not colorable. Suppose c' is a coloring of (F', L'). We construct a coloring of (G, L). Set $c(v) = c'(v)$ for every $v \in V(G) \setminus D$. For $i \in \{1, \ldots, t\}$ and for every $d \in D_i^2$, set $c(d) = c'(d_i^2)$. Then $c(d) \neq c_i$ if $i \leq s$. For $i \in \{1, \ldots, s\}$, set $c(d) = c_i$ for every $d \in D_i^1$, and for $i \in \{s+1, \ldots, t\}$, set $c(d) = c'(d_i^1)$ for every $d \in D_i^1$. Now c is a coloring of (G, L), a contradiction.

Let (F, L') be a minimal obstruction induced by (F', L'). We claim that $V(G) \setminus D \subseteq V(F)$. Suppose not, let $v \in V(G) \setminus (D \cup V(F))$. It follows from the minimality of G that $(G \setminus v, L)$ is colorable. Let c be such a coloring. Set $c'(v) = c(v)$ for every $v \in V(F) \setminus D$. Let $i \in \{1, \ldots, t\}$. If $U_i \neq \{v\}$, then each of these sets U_i, D_i^1, D_i^2 is monochromatic in c. If $U_i = \{v\}$, then D_i is anticomplete to $V(G) \setminus (D_i \cup \{v\})$, and we may assume that each of D_i^1, D_i^2 is monochromatic in c. Now set $c'(d_i^2)$ to be the unique color that appears in D_i^2, and for $i \in \{s+1, \ldots t\}$, set $c'(d_i^1)$ to

be the unique color that appears in D_i^1. Then c' is a coloring of $(F \setminus v, L)$, a contradiction. Now Claim 5 follows from at most $t \leq |V(F)|$ applications of Lemma 4. □

(6) *Let $x \in V(G)$ such that $L(x) = \{1,2,3\}$ and let $U, W \subseteq V(G)$ be disjoint non-empty sets such that $N(x) = U \cup W$ and U is complete to W. Then (possibly exchanging the roles of U and W) either*

1. *there is a path P with $|V(P)| = 4$, such that the ends of P are in U, no internal vertex of P is in U, and $V(P) \cap W = \emptyset$, or*
2. *there exist distinct $i, j \in \{1,2,3\}$ and vertices $u_i, u_j \in U$ and w_i, w_j such that for every $k \in \{i,j\}$*
   - *$k \in L(u_k)$,*
   - *$|L(w_k) \cap \{i,j\}| = 1$,*
   - *there is a path P_k from u_k to w_k,*
   - *if $|V(P_k)|$ is even, then $k \notin L(w_k)$,*
   - *if $|V(P_k)|$ is odd, then $k \in L(w_k)$*

   and $V(P_i)$ is anticomplete to $V(P_j)$.

Proof. Since we may assume that $G \neq K_4$, it follows that U and W are both stable sets. By the minimality of G, there is an L-coloring of $G \setminus x$, say c. Since G does not have an L-coloring, we may assume that there exist $u_1, u_2 \in U$ such that $c(u_i) = i$. Then $c(w) = 3$ for every $w \in W$, and $c(u) \in \{1,2\}$ for every $u \in U$. For $i = 1,2$ let $U_i = \{u \in U : c(u) = i\}$. Let $V_{12} = \{v \in V(G) : c(v) \in \{1,2\}\}$, and let $G_{12} = G|V_{12}$. Suppose first that some component of G_{12} meets both U_1 and U_2. Let P be a shortest path from U_1 to U_2 in G_{12}. Then $|V(P)| = 4$ since G is P_6-free. Moreover, $V(P) \cap W = \emptyset$, and no interior vertex of P is in U, and 6.1 holds.

So we may assume that no component of G_{12} meets both U_1 and U_2. For $i = 1,2$ let V_i be the union of the components of G_{12} that meet U_i. Then V_1 is anticomplete to V_2. If we can exchange the colors 1 and 2 on every component that meets U_1, then doing so produces a coloring of $G \setminus x$ where every vertex of U is colored 2 and every vertex of W is colored 3; this coloring can then be extended to G, which is a contradiction. So there is a component D that meets U_1 and where such

an exchange is not possible. Let (D_1, D_2) be a bipartition of D. We may assume that $U_1 \cap D_2 = \emptyset$. Let us say that $d \in D$ is *deficient* if either $d \in D_1$ and $2 \notin L(d)$, or $d \in D_2$ and $1 \notin L(d)$. Then there is a deficient vertex in D. Let P_1 be a shortest path in D from some vertex $u_1 \in U_1$ to a deficient vertex w_1 of D. Then u_1, w_1, P_1 satisfy the conditions of 6.2. It follows from symmetry that there exist $u_2, w_2 \in V_2$ and a path P_2 from u_2 to w_2 satisfying the condition of 6.2. Since V_1 is anticomplete to V_2, it follows that $V(P_1)$ is anticomplete to $V(P_2)$ and 6.2 holds. □

We also make use of the following result.

4.3.6 (Camby and Schaudt [8]). *For all $t \geq 3$, any connected P_t-free graph H contains a connected dominating set whose induced subgraph is either P_{t-2}-free, or isomorphic to P_{t-2}.*

Our strategy from now on is as follows. Let (G, L) be a minimal list-obstruction, where G is P_6-free. At every step, we find a subgraph R of G, and consider all possible partial precolorings of R. For each precoloring, we update three times with respect to R, and possibly modify the lists further, to produce a minimal list-obstruction (G', L') where G' is an induced subgraph of G, and $|L(v)| \leq 2$ for every $v \in V(G')$, and such that $|V(G)|$ is bounded from above by a function of $|V(G')|$ (the function does not depend on G, it works for all P_6-free graphs G). Since by Lemma 4.3.1 $V(G')$ has bounded size, it follows from Lemma 4.3.3 and Lemma 4.3.4 that $V(G) \setminus R$ has bounded size. Now we use the minimality of G and the internal structure of R to show that R also has a bounded number of vertices, and so $|V(G)|$ is bounded. Next we present the details of the proof.

(1) *There exists an integer C such that the following holds. Let (G, L) is a minimal list-obstruction, where G is C_5-free. Then $|V(G)| \leq C$.*

Proof. We may assume that $|V(G)| > 8$, and therefore there is no K_4 in G. Let H be as in Theorem 4.3.6. Then H is either P_4 or P_4-free. If $|V(H)| \leq 4$, the result follows from Lemma 4.3.5, so we may assume that H is P_4-free. Now by a result of [52] $V(H) = A \cup B$, where A is complete to B, and both A and B are non-empty. Choose $a \in A, b \in B$. Define the set S_0 as follows. If there is a vertex c complete to $\{a, b\}$, let $S_0 = \{a, b, c\}$; if no such c exists, let $S_0 = \{a, b\}$. Then no vertex of $V(G)$ is complete to S_0. Let X_0 be the set of vertices of G with a neighbor in S_0, and let $Y_0 = V(G) \setminus (S_0 \cup X_0)$. By Theorem 4.3.6 every vertex of Y_0 has a neighbor in X_0. By Lemma 4.3.3 and Lemma 4.3.4 we may assume that the vertices of S_0 are precolored; let L_1 be the list system

obtained from L by updating three times with respect to S_0. Then $|L_1(x)| \leq 2$ for every $v \in X_0$. Let $S_1' = \{v \in V(G) : |L(v_1)| = 1\}$, and let S_1 be the connected component of S_1' such that $S_0 \subseteq S_1$. Let X_1 be the set of vertices of $V(G) \setminus S_1$ with a neighbor in S_1, and let $Y_1 = V(G) \setminus (S_1 \cup X_1)$. Then $X_1 \cap S_1' = \emptyset$, and every vertex of Y_1 has a neighbor in X_1. Since $S_0 \subseteq S_1$, no vertex of G is complete to S_1. For $i, j \in \{1, 2, 3\}$ let $X_{ij}^1 = \{x \in X_1 : L_1(x_1) = \{i, j\}\}$.

We now construct the sets S_2, X_2, Y_2. For every $i, j \in \{1, 2, 3\}$ let $U_{i,j}$ be defined as follows. If there is a vertex $u \in X_{ij}^1$ such that there exist $y, z, w \in Y_1$ where $\{y, z, w\}$ is a clique and u has exactly one neighbor in $\{y, z, w\}$, choose such a vertex u with $N(u) \cap Y_1$ maximal, and let $U_{i,j} = \{u\}$. If no such vertex u exists, let $U_{i,j} = \emptyset$. Let $X^{1'}_{i,j}$ be the set of vertices of $X_{i,j}^1$ that are anticomplete to $U_{i,j}$. Let Y_1' be the set of vertices in Y_1 that are anticomplete to $U_{1,2} \cup U_{1,3} \cup U_{2,3}$.

Next we define $V_{i,j}$ for every $i, j \in \{1, 2, 3\}$. If there is a vertex $v \in X^{1'}_{ij}$ such that there exist adjacent $y, z \in Y_1'$ where v has exactly one neighbor in $\{y, z\}$, choose such a vertex v with $N(v) \cap Y_1'$ maximal, and let $V_{i,j} = \{v\}$. If no such vertex v exists, let $V_{i,j} = \emptyset$.

Now let $S_2 = S_1 \cup \bigcup_{i,j \in \{1,2,3\}} (U_{ij} \cup V_{ij})$. Precolor the vertices of S_2. Observe that S_2 is connected. Let L_3 be the list system obtained from L_1 by updating three times. By Lemma 4.3.3 and Lemma 4.3.4 we may assume that (G, L_3) is a minimal list-obstruction. Let $S_3' = \{v \in V(G) : |L(v_1)| = 1\}$, and let S_3 be the connected component of S_3' such that $S_2 \subseteq S_3$. Let X_3 be the set of vertices of $V(G) \setminus S_3$ with a neighbor in S_3, and let $Y_3 = V(G) \setminus (S_3 \cup X_3)$. Then every vertex of Y_3 has a neighbor in X_3. Since $S_1 \subseteq S_3$, no vertex of G is complete to S_3, and for every $x \in X_3$, $|L(x)| = 2$. For $i, j \in \{1, 2, 3\}$ let $X_{ij}^3 = \{x \in X_1 : L_1(x_1) = \{i, j\}\}$. Then X_{ij}^3 is anticomplete to $U_{ij} \cup V_{ij}$.

No vertex of X_3 is mixed on an edge of Y_3. (4.3)

Suppose that there exist $x \in X_3$ and $y, z \in Y_3$ such that y is adjacent to z, and x is adjacent to y and not to z. We may assume that $x \in X_{12}^3$. Then $x \in X_{12}^1 \cup Y_1$, and $y, z \in Y_1$. Suppose first that $x \in Y_1$. Then there is $s_3 \in S_3 \setminus S_1$ such that x is adjacent to s_3. Since $y, z \in Y_3$, it follows that s_3 is anticomplete to $\{y, z\}$. Since $s_3 \in S_3 \setminus S_1$, there is a path P from s_3 to some vertex $s_3' \in S_3$, such s_3' has a neighbor in S_1, and $V(P) \setminus \{s_3'\}$ is anticomplete to S_1. Then s_3' is not complete to S_1, and since S_1 is connected, it follows from Claim 1 that there exist $s_1, s_1' \in S_1$ such that s_3'-s_1-s_1' is a path. But now z-y-x-s_3-P-s_3'-s_1-s_1' is a P_6, a contradiction. This proves that $x \notin Y_1$, and therefore

$x \in X_{12}^1$.

Since $y, z \in Y_1 \cap Y_3$, it follows that $V_{12} \neq \emptyset$. Let v be the unique element of V_{12}. Then v is non-adjacent to x, y, z. Since x is adjacent to y, and v is non-adjacent to y, it follows from the choice of v that there exists $y' \in Y_1$ such that y' is adjacent to v and not to x. Since v has a neighbor in S_1, and v is not complete to S_1, and since S_1 is connected, Claim 1 implies that there exist $s, s' \in S_1$ such that v-s-s' is a path. Since neither of s'-s-v-y'-y-z and s'-s-v-y'-z-y is a P_6, it follows that y' is either complete or anticomplete to $\{y, z\}$. Suppose first that y' is anticomplete to $\{y, z\}$. Let P be a path from v to x with interior in S_1. Then $|V(P)| \geq 3$. Now y'-v-P-x-y-z is a P_6, a contradiction. This proves that y' is complete to $\{y, z\}$.

Now $\{y', y, z\}$ is a clique in Y_1, and v has exactly one neighbor in it. This implies that $U_{12} \neq \emptyset$. Let u be the unique element of U_{12}. Then u is anticomplete to $\{v, y', y, z\}$. It follows from the maximality of u that there exists $y'' \in Y_1$ such that y'' is adjacent to u and non-adjacent to v. Since u has a neighbor in S_1, and u is not complete to S_1, and since S_1 is connected, Claim 1 implies that there exist $t, t' \in S_1$ such that $u - t - t'$ is a path. Suppose y'' has a neighbor in $\{y', y, z\}$. Since $G \neq K_4$, there exist $q, q' \in \{y', y, z\}$ such that y'' is adjacent to q and not to q'. But now t'-t-u-y''-q-q' is a P_6, a contradiction. This proves that y'' anticomplete to $\{y', y, z\}$. Let P be a path from u to v with interior in S_1. Then $|V(P)| \geq 3$. Now y''-v-P-u-y'-y is a P_6, a contradiction. This proves (4.3).

For $i, j \in \{1, 2, 3\}$ let X_{ij} be the set of vertices in X_{ij}^3 with a neighbor in Y_3.

The sets X_{12}, X_{13}, X_{23} are pairwise complete to each other. (4.4)

Suppose $x_1 \in X_{12}$ is non-adjacent to $x_2 \in X_{13}$. Since S_3 is connected and both x_1, x_2 have neighbors in S_3, there is a path P from x_1 to x_2 with $V(P) \setminus \{x_1, x_2\} \subseteq S_3$. Since $L_3(x_1) = \{1, 2\}$ and $L_3(x_2) = \{1, 3\}$, it follows that no vertex of S_3 is adjacent to both x_1 and x_2, and so $|V(P)| \geq 4$. Let $y_i \in Y_3$ be adjacent to x_i. If $|V(P)| > 4$ or $y_1 \neq y_2$, then y_1-x_1-P-x_2-y_2 contains a path with at least six vertices, a contradiction. So $|V(P)| = 4$ and $y_1 = y_2$. But now y_1-x_1-P-x_2-y_1 is a C_5 in G, again a contradiction. This proves (4.4).

Let $D_1, \ldots, D_t$ be the components of Y_3 that have size at least two. Moreover, let $Y' = \bigcup_{i=1}^{t} D_i$.

There is an induced subgraph F of G with $V(G) \setminus Y' \subseteq V(F)$ and a list system L' such that

- $|L'(v)| \leq 2$ *for every* $v \in V(F) \cap Y'$,
- $L'(v) = L(v)$ *for every* $v \in V(F) \setminus Y'$, *and*
- (F, L') *is a minimal list-obstruction, and* $|V(G)|$ *depends only on* $|V(F)|$.

(4.5)

For $in \in \{1, \ldots, t\}$ let U_i be the set of vertices of X_3 with a neighbor in D_i. It follows from Claim 1 and (4.3) that U_i is complete to D_i. Since each D_i contains an edge, (4.4) implies that each U_i is a subset of one of $X^3_{1,3}, X^3_{1,3}, X^3_{2,3}$. Therefore there exists $c_i \in \{1, 2, 3\}$ such that for every $u \in U_i$, $c_i \notin L(u)$. Now (4.5) follows from Claim 5. This proves (4.5).

Let (F, L') be as in (4.5). Since our goal is to prove that (G, L_3) induces a minimal obstruction of bounded size, it is enough to show that $|V(F)|$ has bounded size (where the bound is independent of G). Therefore we may assume that $G = F$ and $L_3 = L'$, and in particular that $|L_3(v)| \leq 2$ for every $v \in Y'$.

Let $Y = Y_3 \setminus Y'$. Then the set Y is stable, $N(y) \subseteq X_{12} \cup X_{13} \cup X_{23}$ for every $y \in Y$, and for $v \in V(G)$, if $|L_3(v)| = 3$, then $v \in Y$. Moreover, if $y \in Y$ has $|L_3(y)| = 3$ and $N(y) \subseteq X_{ij}$ for some $i, j \in \{1, 2, 3\}$, then (G, L_3) is colorable if and only if $(G \setminus y, L_3)$ is colorable, contrary to the fact that (G, L_3) is a minimal list obstruction. Thus for every $y \in Y$ with $|L_3(y)| = 3$, $N(y)$ meets at least two of X_{12}, X_{13}, X_{23}. By (4.4) it follows that the sets X_{12}, X_{13}, X_{23} are pairwise complete to each other, and therefore no $v \in Y$ has neighbors in all three of X_{12}, X_{13}, X_{23}.

Next we define a refinement $\mathcal{L}$ of L_3.

- If exactly one of X_{12}, X_{13}, X_{23} is non-empty, then $\mathcal{L} = \{L_3\}$.
- If at least two of X_{12}, X_{13}, X_{23} are non-empty and some X_{ij} contains two adjacent vertices a, b, let L' be the list obtained by precoloring $\{a, b\}$ and updating three times, and let $\mathcal{L} = \{L'\}$.
- Now assume that at least two of X_{ij} are non-empty, and each of X_{ij} is a stable set. Observe that in this case, in every coloring of G at least one of X_{ij} is monochromatic. For all i, j such that $X_{ij} \neq \emptyset$ and for all $k \in \{i, j\}$ add to $\mathcal{L}$ the list system L^i_{ij}, where $L^i_{ij}(x) = \{i\}$ for all

$x \in X_{ij}$ and $L^i_{ij}(v) = L_3(v)$ for all $v \in V(G) \setminus X_{ij}$, and we updated three times with respect to X_{ij}.

Now $\mathcal{L}$ is a refinement of L and satisfies the hypotheses of Lemma 4.3.3. We claim that for every $L' \in \mathcal{L}$ there exist $i, j \in \{1, 2, 3\}$ such that after the first step of updating $|L'(x)| = 1$ for all $x \in (X_{12} \cup X_{13} \cup X_{23}) \setminus X_{ij}$,

In view of (4.4), this is clear if some X_{ij} is not stable or if only one of the sets X_{12}, X_{13}, X_{23} is non-empty. So we may assume that all X_{ij} are stable, and at least two are non-empty. Let $L' = L^i_{ij}$. Then $L'(x) = \{i\}$ for all $x \in X_{ij}$. Let $k \in \{1, 2, 3\} \setminus \{i, j\}$, then by (4.4) after the first step of updating $L'(x) = \{k\}$ for every $x \in X_{ik}$. Thus after the first step of updating only one of the sets X_{ij} may contain vertices with lists of size two.

Since every $y \in Y$ with $|L_3(y)| = 3$ has neighbors in at least two of X_{12}, X_{13}, X_{23}, it follows that after the second step of updating all vertices of Y have lists of size at most two, and so for all $L' \in \mathcal{L}$ we have that $|L'(v)| \leq 2$ for all $v \in V(G)$. By Lemma 4.3.1, each of (G, L') induces a minimal obstruction with at most 100 vertices. Applying the Lemma 4.3.3 and Lemma 4.3.4, we deduce that $|V(G) \setminus (X_{12} \cup X_{13} \cup X_{23})|$ depends only on the sizes of the minimal obstructions induced by (G, L'), and therefore does not depend on G. Now, since each of X_{ij} is a stable set, Claim 3 implies that $|V(G)|$ is bounded, and Claim 1 follows. □

In the remainder of this proof we deal with minimal list-obstructions (G, L) containing a C_5, by taking advantage of the structure that it imposes. Let C be a C_5 in G, say $C = c_1$-c_2-c_3-c_4-c_5-c_1. Let $X(C)$ be the set of vertices of $V(G) \setminus V(C)$ that have a neighbor in C, let $Y(C)$ be the set of vertices of $V(G) \setminus (V(C) \cup X(C))$ that have a neighbor in X, and let $Z(C) = V(G) \setminus (V(C) \cup X(C) \cup Y(C))$.

(2) *Assume that $|V(G)| \geq 7$. Then the following assertions hold.*

1. *For every $x \in X(C)$ there exist indices $i, j \in \{1, \ldots, 5\}$ such that x-c_i-c_j is an induced path.*
2. *No vertex of $Y(C)$ is mixed on an edge of $G|Z(C)$.*
3. *If $v \in X(C)$ is mixed on an edge of $G|(Y(C) \cup Z(C))$, then the set of neighbors of v in C is not contained in a 3-vertex path of C.*
4. *If $v \in X(C)$ has a neighbor in $Y(C)$, then the set of neighbors of v in C is not contained in a 2-vertex path of C.*

5. *If $z \in Z(C)$ and $u, t \in N(z) \cap Y(C)$ are non-adjacent, then no vertex of $X(C)$ is mixed on $\{u, t\}$.*

6. *Let D be a component of $Z(C)$ with $|D| = 1$, and let N be the set of vertices of $Y(C)$ with a neighbor in D. Then either N is anticonnected, or $N = U \cup W$ where U and W are stable sets, and U is complete to W.*

7. *Let D be a component of $Z(C)$ with $|D| > 1$, and let N be the set of vertices of $Y(C)$ with a neighbor in D. Then D is bipartite, and N is a stable set complete to D.*

Proof. Since $|V(G)| \geq 7$, no vertex is complete to $V(C)$, as that would lead to a list-obstruction on 6 vertices. Thus, the first assertion follows from the fact that G is connected and Claim 1.

Next we prove the second assertion. Suppose that $u \in Y(C)$ is mixed on the edge st with $s, t \in Z(C)$, namely u is adjacent to s and not to t. Let $b \in N(u) \cap X$ and $i, j \in \{1, \ldots, 5\}$ be such that b-c_i-c_j is an induced path (as in Claim 2.1). Then t-s-u-b-c_i-c_j is a P_6, a contradiction.

To see the third assertion, suppose that $x \in X$ is adjacent to $t \in Y$ and non-adjacent to $s \in Y \cup Z$, where t is adjacent to s, and suppose that $N(x) \cap V(C) \subseteq \{c_1, c_2, c_3\}$. We may assume that x is adjacent to c_3. Then c_5-c_4-c_3-x-t-s is a P_6 in G, a contradiction.

To prove the fourth statement, we may assume that $x \in X$ is adjacent to c_1 and to $y \in Y$, and non-adjacent to c_2, c_3 and c_4. Now y-x-c_1-c_2-c_3-c_4 is a P_6 in G, a contradiction.

To prove the fifth statement, suppose that $w \in X(C)$ is adjacent to u and non-adjacent to t. By Claim 2.1 there exists i, j such that w-c_i-c_j is an induced path. Then t-z-u-w-c_i-c_{i+1} is a P_6, a contradiction.

Next let $D = \{v\}$ be a component of Z, then $N(v) \subseteq Y$, and Claim 2.6 follows immediately from the fact that there is no K_4 in G.

Finally let D be a component of $Z(C)$ with $|D| > 1$. By Claim 2.2 N is complete to D. Since there is no K_4 in G, it follows that D is bipartite and N is a stable set. This proves Claim 2.7. □

By Lemma 4.3.3 and Lemma 4.3.4 we may assume that in (G, L) the vertices of C are precolored, and that we have updated three times with respect to $V(C)$. We may assume that $|V(G)| > 8$.

(3) *There is an induced subgraph F of G with $V(G) \setminus Z(C) \subseteq V(F)$ and a list system L' such that*

- $|L'(v)| \leq 2$ *for every* $v \in V(F) \cap Z(C)$,

- *$L'(v) = L(v)$ for every $v \in V(F) \setminus Z(C)$, and*
- *(F, L') is a minimal list-obstruction, and $|V(G)|$ depends only on the size of $|V(F)|$.*

Proof. We write $X = X(C)$, $Y = Y(C)$ and $Z = Z(C)$. Let $D_1, \ldots, D_t$ be components of Z with $|D_i| \geq 2$. Write $D = \bigcup_{i=1}^{t} D_i$. For every i let U_i be the set of vertices of Y with a neighbor in D_i. By Claim 2.7 for every i, U_i is a stable set complete to D_i. By Claim 2.3 every $x \in X$ with a neighbor in U_i has neighbors of two different colors in $V(C)$, and so every such x has list of size one after the first step of updating. Now by Claim 2.5 and since we have updated three times, it follows that for every i there exists $c_i \in \{1, 2, 3\}$ such that for every $u \in U_i$, $c_i \notin L(u)$. By Claim 5 there exist an induced subgraph F of G with $V(G) \setminus Z(C) \subseteq V(F)$ and a list system L' such that

- $|L'(v)| \leq 2$ for every $v \in V(F) \cap D$,
- $L'(v) = L(v)$ for every $v \in V(F) \setminus Z(C)$, and
- (F, L') is a minimal list-obstruction, and $|V(G)|$ depends only on the size of $|V(F)|$.

It remains to show that that $L'(v) \leq 2$ for every $v \in V(F) \cap Z$. Suppose there is $v \in V(F) \cap Z$ with $|L(v)| = 3$. Let D be the component of Z containing v. Then $D = \{v\}$. If $N(v)$ is anticonnected, then by Claim 2.5 every $x \in X$ with a neighbor in $N(v)$ dominates v, contrary to the fact that (F, L') is a minimal list-obstruction. So by Claim 2.6 $N(v) = U \cup W$, both U and W are stable sets, and U is complete to W.

We now apply Claim 6. We may assume that if Claim 6.1 holds then $p_1, p_4 \in U$, and if Claim 6.2 holds, then $u_1, u_2 \in U$. We show that in both cases some vertex $t \in V(G) \setminus U$ is mixed on U. If Claim 6.1 holds, we can take $t = p_1$, so we may assume that Claim 6.2 holds. We may assume that $u_1 = w_1$ and $u_2 = w_2$, for otherwise some vertex of $V(P_1) \cup V(P_2)$ is mixed on U. By Claim 2.3 and Claim 2.5, and since we have updated, it follows that there exists $i \in \{1, 2, 3\}$ such that for every $u \in U$, $i \notin L(u)$. Since $|L(v)| = 3$, and we have updated three times, it follows that after the second step of updating all $u \in U$ have exactly the same list, and this list has size two. Since $u_1 = w_1$ and $u_2 = w_2$, it follows that the lists of u_1 and u_2 changed and became different in the third step of updating, and so some vertex $V(G) \setminus U$ is mixed on U, as required. This proves the claim. Let t be a vertex of $V(G) \setminus U$ that is mixed on U. By Claim 2.5, it follows that $t \in Y \cup Z$.

First we show that if $y \in Y \setminus (U \cup W)$ has a neighbor $u \in U$, and $x \in X$ is adjacent to y, then x is complete to U. Suppose not, let i be such that $x - c_i - c_{i+1}$ is a path (such i exists by Claim 2.1). By Claim 2.5, x is anticomplete to U. Then v-u-y-x-c_i-c_{i+1} is a P_6, a contradiction. This proves the claim.

Now we claim that $Y \setminus (U \cup W)$ is anticomplete to $U \cup W$. Suppose $y \in Y$ has a neighbor $u \in U$, and let $x \in X$ be adjacent to y. Then x is complete to U. Since x does not dominate v, it follows that x has a non-neighbor $w \in W$, and again by the previous claim, y is anticomplete to W. Let $x_1 \in X$ be adjacent to w. By Claim 2.5 x_1 is complete to W. Since x_1 does not dominate v, it follows that x_1 has a non-neighbor in U, and so by Claim 2.5 x_1 is anticomplete to U. By the previous claim, x_1 is non-adjacent to y. Let i be such that x_1-c_i-c_{i+1} is a path (such i exists by Claim 2.1). Now c_{i+1}-c_i-x_1-w-u-y is a P_6, a contradiction. This proves the claim, and in particular we deduce that $t \in Z$.

Since t is mixed on U, there exists an edge a, b with one end in U and the other in W, such that t is adjacent to b and not to a. Let $x \in X$ be adjacent to a. Since x does not dominate v, we deduce that x is not complete to $U \cup W$, and so by Claim 2.5 x is non-adjacent to b. Let i be such that x-c_i-c_{i+1} is a path (as in 2.1). Now t-b-a-x-c_i-c_{i+1} is a P_6, a contradiction. This proves Claim 3. □

Let (F, L') be as in Claim 3. Since our goal is to prove that $|V(G)|$ is bounded, it is enough to prove that $|V(F)|$ is bounded, and so we may assume that $G = F$, $L = L'$, and in particular $|L(v)| \leq 2$ for every $v \in Z(C)$.

(4) *Assume that in the precoloring of C c_2 and c_5 receive the same color, say j. Let $A = \{a \in X(C) : N(a) \cap V(C) = \{c_2, c_5\}\}$ and $W = \{y \in Y : N(y) \cap X \subseteq A\}$. Let D be a component of W such that there exists a vertex with list of size 3 in D, and let N be the set of vertices of A with a neighbor in D. Then D is complete to N, and either*

- *D is anticomplete to $V(G) \setminus (D \cup N)$, or*
- *there exists vertices $d \in D$ and $v \in N(d)$ such that precoloring d, v with distinct colors and updating with respect to the set $\{d, v\}$ three times reduces the list size of all vertices in W to at most two.*

Proof. By Claim 2.3 no vertex of $X(C)$ is mixed on D, and so N is complete to D. Also by Claim 2.3 W is anticomplete to $Z(C)$.

Let $d \in D$, and let $v \in N(d) \setminus (N \cup D)$. Since $D \subseteq W$, it follows that $v \in Y(C)$. By Claim 2.3, $N(v) \cap A = N$. Let $x \in N(v) \cap (X(C) \setminus A)$. Then $N(x) \cap V(C)$ are not contained in a 3-vertex path of C, and therefore $|L(x)| = 1$.

First suppose that $N(x) \cap \{c_2, c_5\} \neq \emptyset$. Then $j \notin L(x)$. We precolor $\{v, d\}$ and update with respect to the set $\{v, d\}$ three times. We may assume that the precoloring of $G|(V(C) \cup \{v, d\})$ is proper. Since $\{v, d\}$ is complete to N and not both v, d are precolored j, it follows that $|L(n)| = 1$ for every $n \in N$, and $|L(u)| \leq 2$ for every $u \in W$ such that u has a neighbor N. Suppose there is $t \in W$ with $|L(t)| = 3$. Then t is anticomplete to $\{v, d\}$. Since $t \in W$, there exists $s \in A$ adjacent to t, and so s is not complete to $\{v, d\}$. Since $s \in A$, it follows from Claim 2.3 that s is not mixed on the edge vd, and so s is anticomplete to $\{v, d\}$. Since $|L(t)| = 3$, it follows that $L(s) = \{1, 2, 3\} \setminus \{j\}$, and so s is non-adjacent to x (since we have updated three times with respect to $V(C)$). Assume by symmetry that c_2 is adjacent to x, then t-s-c_2-x-v-d is a P_6, a contradiction.

Therefore we may assume that $N(x) \cap \{c_2, c_5\} = \emptyset$, and so $N(x) = \{c_1, c_3, c_4\}$. It follows that $L(x) = \{j\}$, and consequently $L(v) \subseteq \{1, 2, 3\} \setminus \{j\}$. If $D = \{d\}$, then $|L(d)| = 3$; but $L(u) \subseteq \{1, 2, 3\} \setminus \{j\}$ for all $u \in N(d)$, which contradicts the fact that (G, L) is a minimal list obstruction. Therefore we may assume there exists $d' \in N(d) \cap D$. Since G is not a K_4, d' is not adjacent to v. But now c_5-c_1-x-v-d-d' is a P_6, a contradiction. □

(5) *Assume that there is a vertex $c_1' \in V(G)$ adjacent to c_1, c_2, c_5 and non-adjacent to c_3, c_4. Then $|V(G)|$ is bounded from above (and the bound does not depend on G).*

Proof. By Lemma 4.3.3 and Lemma 4.3.4, we can precolor the vertices of $V(C) \cup \{c_1'\}$ and update with respect to $V(C) \cup \{c_1'\}$ three times. By symmetry, we may assume that $L(c_1) = \{1\}$, $L(c_1') = L(c_3) = \{2\}$, $L(c_2) = L(c_5) = \{3\}$ and $L(c_4) = \{1\}$. Let $C' = c_1'$-c_2-c_3-c_4-c_5-c_1'. We write $X = X(C)$, $X' = X(C')$, and define the sets Y, Y', Z, and Z' in a similar manner. We abuse notation an denote the list system thus obtained by L. Recall that (G, L) is a minimal list-obstruction.

Let A be the set of all vertices $a \in X \cup X'$ for which $N(a) \cap \{c_1, c_1', c_2, c_3, c_4, c_5\} = \{c_2, c_5\}$. Let W be the set of vertices $y \in Y \cap Y'$ such that $N(y) \cap (X \cup X') \subseteq A$. Since we have updated $|L(x)| \leq 2$ for every $x \in X \cup X'$. By Claim 3 applied to C', we may assume that $|L(z)| \leq 2$ for

every $z \in Z \cup Z'$. Thus if $|L(v)| = 3$ then $v \in Y \cap Y'$, and an easy case analysis shows that $v \in W$. By Lemma 4.3.1 we may assume that $W \neq \emptyset$. Let $D_1, \ldots, D_t$ be the components of W that contain vertices with lists of size three. Suppose first that $|D_i| = \{d\}$ for some i. Then, letting c be a coloring of $G \setminus d$, we observe that no vertex of $N(d)$ is colored 3, and so we can get a coloring of G by setting $c(d) = 3$, a contradiction. This proves that $|D_i| \geq 2$ for every i.

Let $i \in \{1, \ldots, t\}$. Let U_i be the set of vertices of A with a neighbor in D_i. By Claim 4 , D_i is complete to U_i and anticomplete to $V(G) \setminus (D_i \cup U_i)$. Since $U_i \subseteq A$, it follows that $3 \notin L(u)$ for every $u \in U_i$. Let (F, L') be as in Claim 5. Since $|L'(v)| \leq 2$ for every $v \in V(F)$, Lemma 4.3.1 implies that $|V(F)| \leq 100$. Since $|V(G)|$ depends only on $|V(F)|$, Claim 5 follows. This completes the proof of Claim 5. □

We can now prove the following claim, which is the last step of our argument. We may assume that C is precolored in such a way that the precoloring is proper, and the set $\{c_2, c_4\}$ is monochromatic and the set $\{c_3, c_4\}$ is monochromatic.

(6) $|V(G)|$ *is bounded from above (and the bound does not depend on* G*).*

Proof. We may assume that $L(c_1) = 1$, $L(c_2) = L(c_4) = 2$ and $L(c_3) = L(c_5) = 3$. Write $X = X(C), Y = Y(C)$ and $Z = Z(C)$. Let $A' = \{v \in X : N(v) \cap C = \{c_2, c_4\}\}$ and $B' = \{v \in X : N(v) \cap C = \{c_3, c_5\}\}$.

It follows from Claim 2.4 that after the first step of updating every $v \in X \setminus (A' \cup B')$ with a neighbor in Y has list of size one. Let Y' be the set of vertices that have lists of size 3 after the third step of updating. Since $L(z) \leq 2$ for every $z \in Z$, it follows that $Y' \subseteq Y$, and $N(y) \cap X \subseteq A \cup B$ for every $y \in Y'$.

Let A, B be the subsets of A', B' respectively consisting of all vertices with a neighbor in Y'. Then after the second step of updating, the list of every vertex in A is $\{1, 3\}$ and the list of every vertex in B is $\{1, 2\}$. If one of A', B' is not a stable set, Claim 5 completes the proof. So, we may assume that each of A', B' is a stable set.

Let H be the graph obtained from $G|(A \cup B)$ by making each of A, B a clique. Let $C_1, \ldots, C_t$ be the anticomponents of H such that both $A_i = C_i \cap A$ and $B_i = C_i \cap B$ are nonempty. Let $A'' = A \setminus \bigcup_{i=1}^{t} C_i$ and let $B'' = B_i \setminus \bigcup_{i=1}^{t} C_i$.

Let $v \in Y'$. Then $N(v) \cap A$ is complete to $B' \setminus N(v)$, and $N(v) \cap B$ is complete to $A' \setminus N(v)$. In particular, A is complete to $B' \setminus B$, B is complete to $A' \setminus A$, and v is not mixed on C_i for any i. (4.6)

Suppose this is false. By symmetry, we may assume there exists $w \in A$ non-adjacent to $k \in B'$ such that v is adjacent to w but not to k. Then v-w-c_2-c_1-c_5-k is a P_6 in G, a contradiction. This proves (4.6).

Suppose $v \in Y'$ is adjacent to $y \in V(G) \setminus (A \cup B \cup Y')$. Then precoloring y and v and updating three times reduces the list size of all vertices in Y' to at most two. (4.7)

Since $v \in Y'$, it follows that $N(v) \cap X \subseteq A \cup B$, and therefore $y \notin X$. It follows from Claim 2.3 that $N(v) \cap X$ is complete to $N(v) \setminus X$.

By Claim 4 v has both a neighbor in A and a neighbor in B. We precolor v and y and update three times; denote the new list system by L''. If v and y have the same color, or one of v, y is colored 1, then $L''(u) = \emptyset$ for some vertex $u \in N(v) \cap (A \cup B)$, and (4.7) holds. Thus we may assume that one of v, y is precolored 2, and the other one 3. We claim that, after updating, $|L(x)| = 1$ for every $x \in X$. Recall that even before we precolored v and y we had that $|L(x)| = 1$ for every $x \in X \setminus (A' \cup B')$. Since v and y are colored $2, 3$, and $\{v, y\}$ is complete to $N(v) \cap X$, it follows that $|L(x)| = 1$ for every $x \in N(v) \cap X$. Since both $N(v) \cap A$ and $N(v) \cap B$ are nonempty, $L(x) = \{1\}$ for every $x \in N(v) \cap X$. By (4.6), $N(v) \cap A$ is complete to $B' \setminus N(v)$, and $N(v) \cap B$ is complete to $A' \setminus N(v)$. Since we have updated, $L(a) = \{3\}$ for every $a \in A' \setminus N(v)$ and $L(b) = \{2\}$ for every $b \in B' \setminus N(v)$. Consequently $|L(w)| \leq 2$ for every $w \in Y$. This proves 4.7.

In view of (4.7), Lemma 4.3.3 and Lemma 4.3.4, we may assume that $|L(v)| \leq 2$ for every $v \in Y(C)$ for which $N(v) \setminus (A \cup B) \neq \emptyset$.

Let $T = \{y \in Y : N(y) \subseteq A'' \cup B''\}$. There is collection $\mathcal{L}$ of list systems such that for every $L' \in \mathcal{L}$

- *$|L'(v)| \leq 2$ for every $v \in T$, and*
- *$L'(v) = L(v)$ for every $v \in V(G) \setminus T$,*

(4.8)

For every $L' \in \mathcal{L}$, let $(G_{L'}, L')$ be a minimal list obstruction induced by (G, L'). Then $|V(G)|$ depends only on $|\bigcup_{L' \in \mathcal{L}} V(G_{L'})|$.

Let $y \in T \cap Y'$. First we show that y has a neighbor in A and a neighbor in B. Suppose $N(y) \cap B = \emptyset$. Then, by the remark following (4.7), $N(y) \subseteq A$. But now a coloring of $G \setminus y$ can be extended to a coloring of G by assigning color 2 to y, contrary to the fact (G, L) is a minimal obstruction. This proves that y has a neighbor in A and a neighbor in B. In particular both A'' and B'' are non-empty.

Observe that in every coloring of G either A'' or B'' is monochromatic (since they are complete to each other). Let $\mathcal{L}$ be the following collection of list systems. For each $i \in \bigcap_{a \in A''} L(a)$ we add to $\mathcal{L}$ the list system L', where $L'(a) = \{i\}$ for all $a \in A''$ and $L'(v) = L(v)$ for all $v \in V(G) \setminus A''$; and we update three times with respect to A''. Moreover, for each $j \in \bigcap_{b \in B''} L(b)$ we add to $\mathcal{L}$ the list system L', where $L'(b) = \{j\}$ for all $b \in B''$ and $L'(v) = L(v)$ for all $v \in V(G) \setminus B''$, and we update three times with respect to B''.

Now $\mathcal{L}$ is a refinement of L and satisfies the hypotheses of Lemma 4.3.3 with $R = G|(A'' \cup B'')$. Let $L' \in \mathcal{L}$. Since either $|L(a)| = 1$ for every $a \in A''$, or $|L(b)| = 1$ for every $b \in B''$, and since we have updated three times, we have that $|L'(y)| \leq 2$ for every $y \in T$. Let $(G_{L'}, L')$ be a minimal list-obstruction induced by (G, L').

By Lemma 4.3.3 and Lemma 4.3.4,

$$V(G) = A \cup \bigcup_{L' \in \mathcal{L}} V(G_{L'}).$$

Since A is a stable set, Claim 3 implies that $|A|$ only depends on $|\bigcup_{L' \in \mathcal{L}} V(G_{L'})|$, and (4.8) follows. This proves (4.8).

Let $\mathcal{L}$ be as in (4.8). Since our goal is to prove that G has bounded size, it is enough to show that (G, L') induces a minimal obstruction of bounded size for every $L' \in \mathcal{L}$. Therefore we may

assume that for every $y \in Y'$ there exists an index i such that y is complete to C_i.

$$\text{Let } y_1 \in Y' \text{ and let } C_1 \subseteq N(y_1). \text{ Then we may assume that no vertex of } V(G) \setminus C_1 \text{ is mixed on } A_1 \text{ (and similarly on } B_1). \tag{4.9}$$

Suppose $x \in V(G) \setminus C_1$ is mixed on A_1. Since x is mixed on C_1, and C_1 is an anticomponent of H, there exist $a_1 \in A_1$ and $b_1 \in B_1$ such that a_1b_1 is a non-edge, and x is mixed on this non-edge. Let $a_1' \in A_1$ be such that x is mixed on $\{a_1, a_1'\}$. By Lemma 4.3.3 and Lemma 4.3.4 we can precolor $T = \{x, a_1, a_1', b_1, y_1\}$, and update three times with respect to T. Let Y'' be the set of vertices with lists of size 3 after updating. We claim that $Y'' = \emptyset$. Suppose not and let $v \in Y''$. By the remark following (4.8) there exists an index i such that v is complete to C_i. Then $i \neq 1$. Since $v \in Y''$, $\{a_1, a_1'\}$ is complete to B_i and b_1 is complete to A_i, and we have updated three times with respect to T, it follows that $L(a_1) = L(a_1') = \{3\}$ and $L(b_1) = \{2\}$. Since x has a neighbor in $\{a_1, a_1'\}$ we may assume that $L(x) \neq \{3\}$.

First consider the case that x is adjacent to a_1 and not to b_1. Then x is non-adjacent to a_1'. Choose $a_i \in A_i$. Since x-a_1-y_1-b_1-a_i-v is not a P_6 in G, it follows that x is adjacent to a_i. Since $v \in Y''$, it follows that $L(x) = \{2\}$, and therefore x is anticomplete to B_i. Choose b_i such that a_ib_i is a non-edge, then x-a_i-v-b_i-a_1'-y_1 is a P_6 in G, a contradiction. Therefore x is adjacent to b_1 and not to a_1. Since x-b_1-y_1-a_1-b-v is not a P_6 in G for any $b \in B_i$, it follows that x is complete to B_i, which is a contradiction since $L(x) \neq \{3\}$ and $v \in Y''$. This proves (4.9).

$$\text{Let } v \in Y' \text{ and let } C_i \in N(v). \text{ Then we may assume } |A_i| = |B_i| = 1. \tag{4.10}$$

Suppose this is false. We may assume that $i = 1$. By (4.9), no vertex of $G \setminus C_1$ is mixed on A_1 and no vertex of $G \setminus C_1$ is mixed on B_1. Choose $a_1 \in A_1$ and $b_1 \in B_1$ such that a_1b_1 is an edge if possible. Then $(G \setminus (A_1 \cup B_1)) \cup \{a_1, b_1\}$ is not L-colorable, since otherwise we can color A_1 in the color of a_1 and B_1 in the color of b_1. Since (G, L) is a minimal list-obstruction, (4.10) follows.

Let $Y_1 = \{y \in Y' : N(y) \subseteq (A \setminus A'') \cup (B \setminus B'')\}$, and let $Y_2 = Y' \setminus Y_1$. By (4.10) and since (G, L) is a minimal list-obstruction, every $y \in Y_1$ is complete to more than one of $C_1, \ldots, C_t$. We may assume that each of $C_1, \ldots, C_s$ is complete to some vertex of Y_1, and $C_{s+1} \cup \ldots \cup C_t$ is anticomplete to Y_1. Let F be the graph with vertex set $V(F) = \{1, \ldots, s\}$ where i is adjacent to j if and only if there is a vertex $y \in Y_1$ complete to $C_i \cup C_j$. We will refer to the vertices of F as $1, \ldots, s$ and $C_1, \ldots, C_s$ interchangeably.

Let $F_1, \ldots, F_k$ be the components of F, let $A(F_i) = \bigcup_{C_j \in F_i} A_j$, and let $B(F_i) = \bigcup_{C_j \in F_i} B_j$. Moreover, let $Y(F_i) = \{y \in Y_1 : N(y) \subseteq A(F_i) \cup B(F_i)\}$.

> *Let $i \in \{1, \ldots, k\}$ and let $T \subseteq V(G)$ be such that $A(F_i) \cup B(F_i) \cup Y_1 \subseteq T$. Then for every L-coloring of $G|T$, both of the sets $A(F_i)$ and $B(F_i)$ are monochromatic, and the color of $A(F_i)$ is different from the color of $B(F_i)$.* (4.11)

Let c be a coloring of $G|T$. Let $y \in Y(F_i)$. We may assume that y is complete to C_1, and $C_1 \in F_i$. Let $\alpha = c(A_1)$ and $\beta = c(B_1)$, where $c(A_i)$ and $c(B_i)$ denote the color given to the unique vertices in the sets A_i and B_i respectively. Since y is complete to at least two of $C_1, \ldots, C_s$, the sets $N(y) \cap A$ and $N(y) \cap B$ are monochromatic, and $\alpha \neq \beta$. Pick any $t \in F_i$, and let P be a shortest path in F from C_1 to t. Let s be the neighbor of t in P. We may assume that $s = C_2$ and $t = C_3$. We proceed by induction and assume that $c(A_2) = \alpha$, and $c(B_2) = \beta$. Since s is adjacent to t in F, there is $y' \in Y_1$ such that y' is complete to $C_2 \cup C_3$. Then $c(y') \in \{1, 2, 3\} \setminus \{\alpha, \beta\}$. Moreover, A_2 is complete to B_3, and A_3 is complete to B_2, and so $c(A_3) \notin \{c(y'), \beta\}$ and $c(B_3) \notin \{c(y'), \alpha\}$. It follows that $c(A_3) = \alpha$ and $c(B_3) = \beta$, as required. This proves (4.11).

We now construct a new graph G' where we replace each F_i by a representative in A and a representative in B, as follows. Let G' be the graph obtained from $G \setminus (C_1 \cup \ldots \cup C_s \cup Y_1)$ by adding $2s$ new vertices $a_1, \ldots, a_s, b_1, \ldots, b_s$, where

$$N_{G'}(a_i) = \{b_i\} \cup \bigcup_{a \in A(F_i)} (N_G(a) \cap V(G'))$$

and

$$N_{G'}(b_i) = \{a_i\} \cup \bigcup_{b \in B(F_i)} (N_G(b) \cap V(G')),$$

for all $i \in \{1, \ldots, s\}$. Note that, in G', the set $\{a_1, \ldots, a_s\}$ is complete to the set $\{b_1, \ldots, b_s\}$. Let $L(a_i) = \{1, 3\}$ and $L(b_i) = \{1, 2\}$ for every i. By repeated applications of Claim 2, we deduce that G' is P_6-free.

Let $A^* = (A \setminus (A'' \cup A_1 \ldots \cup A_s)) \cup \{a_1, \ldots, a_s\}$ and $B^* = (B \setminus (B'' \cup B_1 \ldots \cup B_s)) \cup \{b_1, \ldots, b_s\}$. Note that A^* is complete to B'', and B^* is complete to A''.

Let $R = G|(A^* \cup B^* \cup A'' \cup B'')$.

We may assume that $|A^*| \geq 2$, and define the list systems L_1, L_2, and L_3 as follows.

$$L_1(v) = \begin{cases} \{3\} & \text{if } v \in A'' \\ \{2\} & \text{if } v \in B'' \\ L(v) & \text{if } v \notin A'' \cup B'' \end{cases}$$

$$L_2(v) = \begin{cases} \{3\} & \text{if } v \in A^* \\ L(v) & \text{if } v \notin A^* \end{cases}$$

$$L_3(v) = \begin{cases} \{2\} & \text{if } v \in B^* \\ L(v) & \text{if } v \notin B^* \end{cases}$$

Let $\mathcal{L} = \{L_1, L_2, L_3\}$. It is clear that, for every L-coloring c of G', there exists a list system $L' \in \mathcal{L}$ such that c is also an L'-coloring of G'. Recall that by the remark following (4.8) every vertex of Y_2 has a neighbor in A^*, a neighbor in B^*, and a neighbor in $A'' \cup B''$. Therefore, for every $L' \in \mathcal{L}$, every vertex in Y_2 is adjacent to some vertex v with $|L'(v)| = 1$. Now by Lemma 4.3.1, Lemma 4.3.3, and Lemma 4.3.4, for every $L' \in \mathcal{L}$, G' contains an induced subgraph G'' such that (G'', L) is not colorable, and $|V(G'') \setminus V(R)| \leq 3 \cdot 36 \cdot 100$. We may assume that for every index i, $a_i \in G''$ or $b_i \in G''$, for otherwise we can just delete F_i from G contradicting the minimality of (G, L).

We claim that the subgraph induced by G on the vertex set

$$S = (V(G) \cap V(G'')) \cup Y_1 \cup \bigcup_{i=1}^{s} (A(F_i) \cup B(F_i))$$

is not L-colorable. Suppose this is false and let c be such a coloring. By (4.11), for every $i \in \{1, \ldots, k\}$ the sets $A(F_i)$ and $B(F_i)$ are both monochromatic, and c can be converted to a coloring of G'' by giving a_i the unique color that appears in $A(F_i)$ and b_i the unique color that appears in $B(F_i)$, a contradiction. Thus $V(G) = S$, and it is sufficient to show that $|Y_1 \cup \bigcup_{i=1}^{s} (A(F_i) \cup B(F_i))|$ has bounded size. To see this, let $T = S \setminus (A \cup B \cup Y_1)$, then $|T| < |V(G'') \setminus R| \leq 3 \cdot 36 \cdot 100$.

First we bound s. Partition the set of pairs $\{(a_1, b_1), \ldots, (a_s, b_s)\}$ according to the adjacency of each (a_i, b_i) in T; let $H_1, \ldots, H_l$ be the blocks of this partition. Then $l \leq 2^{2|T|}$.

We claim that $|H_i| = 1$ for every i. Suppose for a contradiction that $(a_i, b_i), (a_j, b_j) \in H_1$. Let c be an L-coloring of $G'' \setminus \{a_i, b_i\}$. Note that, since $N(a_i) = N(a_j)$ and $N(b_i) = N(b_j)$, setting $c(a_i) = c(a_j)$ and $c(b_i) = c(b_j)$ gives an L-coloring of G'', a contradiction. This proves that $s \leq 2^{2|T|}$.

Next we bound $|F_i|$ for each i. Let $i \in \{1, \ldots, s\}$. Partition the set $\{C_j : j \in F_i\}$ according to the adjacency of C_j in T. Let $C^i_1, \ldots, C^i_{q_i}$ be the blocks of the partition. Then $q_i \leq 2^{|T|}$. Let $C_l \in C^i_1$. For each $j \in \{2, \ldots, q_i\}$ let Q^i_j be a shortest path from C_l to C^i_j in F. In G, Q^i_j yields a path $Q^{i\prime}_j = a'_1$-y'_1-a'_2-y'_2-$\ldots$-y'_m-a'_m where $a'_1 \in C_l$, $a'_m \in A \cap C^i_j$, $a'_2, \ldots, a'_{m-1} \in \bigcup_{l \in \{1,\ldots,q\} \setminus \{1,j\}} A \cap C^i_l$ and $y'_1, \ldots, y'_m \in Y_1$. Let $Y(Q^i_j) = \{y'_1, \ldots, y'_m\}$. Since $Q^{i\prime}_j$ does not contain a P_6, it follows that $|Y(Q_j)| \leq 2$. Let $Y^i_1 = \bigcup_{j=2}^{q_i} Y(Q^i_j)$, and note that $|Y^i_1| \leq 2q_i - 2 \leq 2(2^{|T|} - 1)$. Moreover, let $\hat{Y} = \bigcup_{i=1}^{s} Y^i_1$, and note that $|\hat{Y}| \leq 2(2^{|T|} - 1)s$.

Next we claim that $\hat{Y} = Y_1$. To see this, suppose that there exists a vertex $y \in Y_1 \setminus \hat{Y}$. Note that y is critical, and let c be a coloring of $G \setminus y$. We may assume that $N(y) \subseteq \bigcup_{i \in F_1} C_i$. We will construct a coloring of G'' and obtain a contradiction. By (4.11), for every $i \in \{2, \ldots, s\}$ both of the sets $A(F_i)$ and $B(F_i)$ are monochromatic and so we can color a_i and b_i with the corresponding colors.

Let F' be the graph with vertex set F_1 and such that i is adjacent to j if and only if there is a vertex $y' \in \hat{Y}$ (and therefore $y' \in Y^1_1$) complete to $C_i \cup C_j$. Recall the partition $C^1_1, \ldots, C^1_{q_1}$. By the definition of Y^1_1, there exists $C'_1 \in C^1_1$ such that for every $i \in \{2, \ldots, q_1\}$ there is a path in F' from C'_1 to a member C'_i of C^1_i. Write $\{a'_1\} = C'_1 \cap A$ and $\{b'_1\} = C'_1 \cap B$, and let $\alpha = c(a'_1)$ and $\beta = c(b'_i)$. Following the outline of the proof of (4.11) we deduce that $\alpha \neq \beta$, and that for each $i \in \{1, \ldots, q\}$ some vertex of $\bigcup_{C \in C^1_i} C \cap A$ is colored with color α, and some vertex of $\bigcup_{C \in C^1_i} C \cap B$ is colored with color β. Observe that for every index i only vertices of $Y_1 \cup \bigcup_{C \in C^1_i}(C \cap B)$ are mixed on $\bigcup_{C \in C^1_i}(C \cap A)$, and only vertices of $Y_1 \cup \bigcup_{C \in C^1_i}(C \cap A)$ are mixed on $\bigcup_{C \in C^1_i}(C \cap B)$. Thus we can color a_1 with color α and b_1 with color β, obtaining a coloring of G'', a contradiction. This proves that $|Y_1| \leq 2(2^{|T|} - 1)s$. Now applying Claim 4 $|Y_1|$ times implies that there is a function q that does not depend on G, such that $|\bigcup_{i=1}^{s}(A(F_i) \cup B(F_i))| \leq q(|T|)$. Consequently, $|V(G)| \leq |T| + |Y_1| + q(|T|) \leq |T| + 2(2^{|T|} - 1)s + q(T)$. This completes the proof. □

Now Lemma 4.3.2 follows from Claim 6.

4.4 $2P_3$-free 4-vertex critical graphs

The aim of this section is to show that there are only finitely many $2P_3$-free 4-vertex critical graphs. The proof follows the same outline as the proof of the previous section. Lemma 4.4.1 deals with $2P_3$-free minimal list-obstructions where every list has size at most two. In view of Lemma 4.6.2 the exact analogue of Lemma 4.3.1 does not hold in this case, however if we add the additional assumption that the minimal list-obstruction is contained in a $2P_3$-free 4-vertex-critical graph that was obtained by updating with respect to a set of precolored vertices, then we can show that the size of the obstruction is bounded.

4.4.1. *There is a an integer $C > 0$ such that the following holds. Let (G, L) be a list-obstruction. Assume that G is $2P_3$-free and the following holds.*

(a) Every list contains at most two entries.

(b) Every vertex v of G with $|L(v)| = 2$ has a neighbor u with $|L(u)| = 1$ such that for all $w \in V(G)$ with $|L(w)| = 2$, $uw \in E(G)$ implies $L(w) = L(v)$.

Then (G, L) contains a minimal list-obstruction with at most C vertices.

Like in the case of P_6-free list-obstructions, we can use the precoloring technique to prove that the lemma above implies our main lemma.

4.4.2. *There is an integer $C > 0$ such that every $2P_3$-free 4-vertex-critical graph has at most C vertices. Consequently, there are only finitely many $2P_3$-free 4-vertex-critical graphs.*

4.4.1 Proof of Lemma 4.4.1

Let G' be an induced subgraph of G such that (G', L) is a minimal list-obstruction. By Lemma 4.2.1, it suffices to prove that the length of any propagation path of (G', L) is bounded by a constant. To see this, let $P = v_1$-v_2-…-v_n be a propagation path of (G', L) starting with color α, say. Consider v_1 to be colored with α, and update along P until every vertex of P is colored. Let this coloring of P be denoted by c. Recall condition (4.1) from the definition of propagation path: every edge $v_i v_j$ with $3 \leq i < j \leq n$ and $i \leq j - 2$ is such that

$$S(v_i) = \alpha\beta \text{ and } S(v_j) = \beta\gamma,$$

where $\{1,2,3\} = \{\alpha, \beta, \gamma\}$.

First we prove that there is a constant δ such that there is a subpath $Q = v_m$-v_{m+1}-...-$v_{m'}$ of P of length at least $\lfloor \delta n \rfloor$ with the following property. After permuting colors if necessary, it holds for all $i \in \{m, \ldots, m'\}$ that

$$S(v_i) = \begin{cases} 32, & \text{if } i \equiv 0 \mod 3 \\ 13, & \text{if } i \equiv 1 \mod 3 \\ 21, & \text{if } i \equiv 2 \mod 3 \end{cases}.$$

To see this, suppose there are two indices $i, j \in \{3, \ldots, n-3\}$ such that $i+2 \leq j$ and $c(v_i) = c(v_{i+2}) = c(v_j) = c(v_{j+2})$. Moreover, suppose that $c(v_i) = c(v_{i+2}) = c(v_j) = c(v_{j+2}) = \alpha$ and $c(v_{i+1}) = c(v_{j+1}) = \beta$ for some α, β with $\{\alpha, \beta, \gamma\} = \{1,2,3\}$. Thus, $L(v_{i+1}) = L(v_{i+2}) = L(v_{j+1}) = L(v_{j+2}) = \{\alpha, \beta\}$, $\alpha \in L(v_{j+3})$, and $\alpha \neq c(v_{j+3})$. But now v_i-v_{i+1}-v_{i+2} and v_{j+1}-v_{j+2}-v_{j+3} are both induced P_3's, according to (4.1), and there cannot be any edge between them. This is a contradiction to the assumption that G is $2P_3$-free. The same conclusion holds if $c(v_{i+1}) = c(v_{j+1}) = \gamma$. Hence, there cannot be three indices $i, j, k \in \{3, \ldots, n-3\}$ such that $i+2 \leq j$, $j+2 \leq k$, and

$$c(v_i) = c(v_{i+2}) = c(v_j) = c(v_{j+2}) = c(v_k) = c(v_{k+2}) = \alpha.$$

Consider the following procedure. Pick the smallest index $i \in \{3, \ldots, n-3\}$ such that $c(v_i) = c(v_{i+2}) = 1$, if possible, and remove the vertices v_i, v_{i+1}, and v_{i+2} from P. Let P' be the longer of the two paths v_1-v_2-...-v_{i-1} and v_{i+3}-v_2-...-v_n. Repeat the deletion process and let $P'' = v_r$-v_{r+1}-...-$v_{r'}$ be the path obtained. As shown above, we now know that there is no index $j \in \{r+2, \ldots, r'-3\}$ with $c(v_j) = c(v_{j+2}) = 1$.

Repeating this process for colors 2 and 3 shows that there is some $\delta > 0$ such that there is a path $Q = v_m$-v_{m+1}-...-$v_{m'}$ of length $\lfloor \delta n \rfloor$ where $c(v_i) \neq c(v_{i+2})$ for all $i \in \{m-1, \ldots, m'-2\}$. Thus, after swapping colors if necessary we have the desired property defined above.

From now on we assume that G has sufficiently many vertices and hence $m' - m$ is sufficiently large. Since G is $2P_3$-free and hence P_7-free, the diameter of every connected induced subgraph of G is bounded by a constant. In particular, the diameter of the graph $G|(\{v_m, \ldots, v_{m'}\})$ is bounded, and so we may assume that there is a vertex v_i with $m \leq i \leq m'$ with at least 20 neighbors in the path Q. We may assume that $c(v_i) = 1$ and, thus, $S(v_i) = 13$.

We discuss the case when $|N(v_i) \cap \{v_m, \ldots, v_{i-1}\}| \geq 10$. The case of $|N(v_i) \cap \{v_{i+1}, \ldots, v_{m'}\}| \geq 10$ can be dealt with in complete analogy.

We pick distinct vertices $v_{i_1}, \ldots, v_{i_{10}} \in N(v_i) \cap \{v_m, \ldots, v_{i-1}\}$ where $i_1 < i_2 < \ldots < i_{10}$. Note that (4.1) implies that $S(v_{i_j}) = 21$ for all $j \in \{1, \ldots, 10\}$.

We can pick three indices j_1, j_2, j_3 with $r' < j_1 < j_2 < j_3 < m'$ such that

- $S(v_{j_1}) = S(v_{j_2}) = S(v_{j_3}) = 32$, and
- $i_2 + 5 = j_1$, $j_1 + 6 = j_2$, $j_2 + 4 \leq i_7$, $i_8 + 5 = j_3$, and $j_3 + 4 = i$.

Recall that assumption (b) of the lemma we are proving implies the following. Since $L(v_{j_u}) = \{2, 3\}$, v_{j_u} has a neighbor x_{j_u} with $L(x_{j_u}) = \{1\}$, $u = 1, 2, 3$, such that x_{j_u} is not adjacent to any vertex v_j with $m \leq j \leq m'$ and $j \equiv 1 \mod 3$ or $j \equiv 2 \mod 3$.

Suppose that $x_{j_u} = x_{j_{u'}}$ for some $v_{j_{u'}}$ with $u' \in \{1, 2, 3\} \setminus \{u\}$. Now the path v_{j_u}-x_{j_u}-$v_{j_{u'}}$ is an induced P_3, and so is the path v_{i_1}-v_i-v_{i_2}, both according to condition (4.1). Moreover, there is no edge between those two paths, due to (4.1), which is a contradiction. Hence, the three vertices x_{j_u}, x_{j_u}, and x_{j_u} are mutually distinct and, due to the minimality of (G, L), mutually non-adjacent.

Consider the induced P_3's v_{j_1+1}-v_{j_1}-x_{j_1} and v_{j_3+1}-v_{j_3}-x_{j_3}. Since G is $2P_3$-free, there must be an edge between these two paths. According to (4.1), it must be the edge $v_{j_1+1}v_{j_3}$. For similar reasons, the edge $v_{j_2+1}v_{j_3}$ must be present. Now the path v_{j_1+1}-v_{j_3}-v_{j_2+1} is an induced P_3, and so is the path v_{i_7}-v_i-v_{i_8}. Moreover, there is no edge between those two paths, due to (4.1), which is a contradiction. This completes the proof.

4.4.2 Proof of Lemma 4.4.2

We start with two statements that allow us to precolor sets of vertices with certain properties. In this subsection G is always a $2P_3$-free graph, and all lists are subsets of $\{1, 2, 3\}$.

(1) *Assume that (G, L) is a list-obstruction. Let $X \subseteq V(G)$ be such that there exists a coloring c of $G|X$ with the following property: for each $x \in X$ there exists a set $N_x \subseteq V(G)$ with $|N_x| \leq k$ such that x is colored $c(x)$ in every coloring of $(G|(\{x\} \cup N_x), L)$. Let L' be a list system such that*

$$L'(v) = \begin{cases} L(v), & \text{if } v \in V(G) \setminus X \\ \{c(x)\}, & \text{if } v \in X \end{cases}.$$

Then the following holds.

(a) (G, L') *is a list-obstruction.*

(b) If $K \subseteq V(G)$ *is such that* $(G|K, L')$ *is a minimal list-obstruction induced by* (G, L')*, then* (G, L) *contains a minimal list-obstruction of size at most* $(k+1)|K|$*.*

Proof. Since $L'(v) \subseteq L(v)$ for all $v \in V(G)$, (G, L') is also a list-obstruction. This proves (a).

Let $A = G|(K \cup \bigcup_{x \in K \cap X} N_x)$, then $|V(A)| \leq (k+1)|K|$. Suppose that there exists a coloring, c' of (A, L). Note that for every $x \in V(A)$, $N_x \subseteq A$. Hence by the definition of X, $c'(x) = c(x)$ for every $x \in V(A)$. This implies that c' is also a coloring of (A, L'), which gives a coloring of $(G|K, L')$, a contradiction. Therefore (A, L) is a list-obstruction induced by (G, L). Since $|V(A)| \leq (k+1)|K|$, (b) holds. This completes the proof. □

(2) *Let* (G, L) *be a list-obstruction, and let* $X \subseteq V(G)$ *be a vertex subset such that* $|L(x)| = 1$ *for every* $x \in X$*. Let* $Y = N(X)$*, and let* $Y' \subseteq Y$ *be such that for every* $v \in Y'$*,* $|L(v)| = 3$*. For every* $v \in Y'$*, pick* $x_v \in N(v) \cap X$*. Let* L' *be the list defined as follows.*

$$L'(v) = \begin{cases} L(v), & \text{if } v \in V(G) \setminus Y' \\ L(v) \setminus L(x_v), & \text{if } v \in Y' \end{cases}.$$

Let (G', L') *be a minimal list-obstruction induced by* (G, L')*. Then there exists a minimal list-obstruction induced by* (G, L)*, say* (G'', L)*, with* $|V(G'')| \leq 2|V(G')|$*.*

Proof. Let $R = \{x_v : v \in V(G') \cap Y'\}$ and let $P = R \cup V(G')$. It follows that $|V(P)| \leq 2|V(G')|$. It remains to prove that $(G|P, L)$ is not colorable. Suppose there exists a coloring c of $(G|P, L)$. Note that c is not a coloring of (G', L') and G' is an induced subgraph of $G|P$. Hence there exists $w \in V(G')$ such that $c(w) \notin L'(w)$. By the construction of L', it follows that $w \in Y'$ and that $c(w) \in L(w) \setminus L'(w) = \{c(x_w)\}$, which is a contradiction. This completes the proof. □

Let G be a $2P_3$-free 4-vertex-critical graph such that $|V(G)| \geq 5$, then the following claim holds.

(3) *At least one of the following holds*

1. *There exists* $S_0 \subseteq V(G)$ *such that* $|S_0| \leq 5$*,* $G|S_0$ *contains a copy of* P_3 *and* $S_0 \cup N(S_0) \cup N(N(S_0)) = V(G)$*, or*

2. G has a semi-dominating set of size at most 5.

Proof. Since G is $2P_3$-free and thus also P_7-free, Theorem 4.3.6 states that G has a dominating induced P_5 or a dominating P_5-free connected induced subgraph, denoted by D_f. Recall that a dominating set is always a semi-dominating set; so we may assume that the latter case holds and $|V(D_f)| \geq 6$. By applying Theorem 4.3.6 to D_f again, we deduce that D_f has a dominating induced subgraph T, which is isomorphic to P_3 or a connected P_3-free graph.

If T is isomorphic to P_3, then we are done by setting $S_0 = V(T)$. Hence we may assume T is a connected P_3-free graph. Therefore T is a complete graph, and so $V(T) \leq 3$. If there exists a vertex $s' \in V(G \setminus T)$ mixed on T, we are done by setting $S_0 = V(T) \cup \{s'\}$. Hence we may assume that for every $v \in V(D_f \setminus T)$, v is complete to T. Since $|V(G)| \geq 5$, it follows that D_f is K_4-free. Therefore there exist $v, w \in V(D_f \setminus T)$ such that v is non-adjacent to w and we are done by setting $S_0 = V(T) \cup \{v, w\}$. □

If G has a semi-dominating set of size at most 5, we are done by Lemma 4.3.5 and Lemma 4.4.1. Hence we may assume there exists S_0 defined as in Claim 3.

For a list system L' of G, we say that (X_1, X_2, B, S) is the *partition with respect to* L' by setting:

(a) $S = \{v \in V(G) : |L'(v)| = 1\}$.

(b) $B = N(S)$; assume that $|L'(v)| = 2$ for every $v \in B$.

(c) Let $X = V(G) \setminus (S \cup B)$. We say that C is a *good component* of X if there exist $x \in C$ and $\{i, j\} \subseteq \{1, 2, 3\}$ so that x has two adjacent neighbors $a, b \in B_{ij}$, where $B_{ij} = \{b \in B$ such that $L'(b) = \{i, j\}\}$. Let X_1 be the union of all good components of X and let $X_2 = X \setminus X_1$.

Let (X_1, X_2, B, S) be the partition with respect to L'. Define $X = X_1 \cup X_2$. For every $1 \leq i \leq j \leq 3$, define $B_{ij} = \{b \in B$ such that $L'(b) = \{i, j\}\}$ and $X_{ij} = \{x \in X_2$ such that $|N(x) \cap B_{ij}| \geq 2\}$. For $\{i, j, k\} = \{1, 2, 3\}$, let us say that a component C of X_2 is *i-wide* if there exist a_j in B_{ik} and a_k in B_{ij} such that C is complete to $\{a_j, a_k\}$. We call a_j and a_k *i-anchors* of C. Note that a component can be i-wide for several values of i. Let L'' be a subsystem of L' and let (X_1', X_2', B', S') be the partition with respect to L''. Then $S \subseteq S'$, $B' \setminus B \subseteq X_1 \cup X_2$ and $X_2' \subseteq X_2$.

Next we define a sequence of new lists $L_0, \ldots, L_5$. Let $\{i, j, k\} = \{1, 2, 3\}$. Let S_0 be as in Claim 3, and let $L_0 = L$.

1. Let L_1 be the list system obtained by precoloring S_0 and updating three times. Let (X_1^1, X_2^1, B^1, S^1) be the partition with respect to L_1.

2. For each $k \in \{1,2,3\}$, choose $x_k \in X_{ij}^1$ such that $|N(x_k) \cap B_{ij}^1|$ is minimum. Let $a_k, b_k \in N(x_k) \cap B_{ij}^1$. Let L_2 be the list system obtained from L_1 by precoloring $\bigcup_{i=1}^{3} \{a_i, b_i, x_i\}$ and updating the lists of vertices three times. Let (X_1^2, X_2^2, B^2, S^2) be the partition with respect to L_2 .

3. For each $k \in \{1,2,3\}$, let $\hat{B}_k \subseteq B_{ij}^2$ with $|\hat{B}_k| \leq 1$ be defined as follows. If there does not exist a vertex $v \in B_{ij}^2$ that starts a path v-u-w where $u, w \in X_2^2$, then $\hat{B}_k = \emptyset$. Otherwise choose $b_k \in B_{ij}^2$ maximizing the number of pairs (u, w) where b_k-u-w is a path and let $\hat{B}_k = \{b_k\}$. Let L_3 be the list system from L_2 obtained by precoloring $\hat{B}_1 \cup \hat{B}_2 \cup \hat{B}_3$ and updating three times. Let (X_1^3, X_2^3, B^3, S^3) be the partition with respect to L_3.

4. Apply step 2 to (X_1^3, X_2^3, B^3, S^3) with list system L_3; let L_4 be the list system obtained and let (X_1^4, X_2^4, B^4, S^4) be the partition with respect to L_4.

5. For every component C_t of X_2^4 with size 2, if C_t is i-wide with i-anchors a^t, b^t, set $L_5(a^t) = L_5(b^t) = \{i\}$; then let L_5 be the list system after updating with respect to $\bigcup_t \{a^t, b^t\}$ three times. Let (X_1^5, X_2^5, B^5, S^5) be a partition with respect to L_5.

By Lemma 4.3.3 and Lemma 4.3.4, it is enough to prove that (G, L_4) induces a bounded size list-obstruction. To do that, we prove the same for (G, L_5), and then use Claim 1 and Lemma 4.3.4, as we explain in the remainder of this section.

We start with a few technical statements.

(4) *Let $1 \leq m \leq l \leq 5$. Then the following holds.*

1. *For every vertex in $x \in X^m$, $|L_m(x)| = 3$, and every component of X^m is a clique with size at most 3.*

2. *If no vertex of B_{ij}^m is mixed on an edge in $G|X_2^m$, then no vertex of B_{ij}^l is mixed on an edge in $G|X_2^l$.*

3. *If no vertex of X_2^m has two neighbors in B_{ij}^m and no vertex of B^m is mixed on an edge in $G|X_2^m$, then no vertex of X_2^l has two neighbors in B_{ij}^l.*

Proof. By construction, for every vertex in $x \in X^m$, $|L_m(x)| = 3$. Observe that $S_0 \subseteq S^m$. Recall that G is $2P_3$-free and that S_0 contains a P_3. Hence X^m does not contain a P_3, and so every component of X^m is a clique. Since $|V(G)| \geq 5$, it follows that every component of X^m has size at most 3. This proves the first statement.

Let $b \in B^l_{ij}$ be mixed on the edge uv such that $u, v \in X^l_2$. Recall that $X^l_2 \subseteq X^m_2$; thus $\{u, v\} \subseteq X^m_2$. By assumption $b \notin B^m_{ij}$ and hence $b \in X^m$. But now b-u-v is a P_3 in X^m, a contradiction. This proves the second statement.

To prove the last statement, suppose that there exists $y \in X^l_2$ with two neighbors $u, v \in B^l_{23}$. Since $y \in X^l_2$, it follows that u, v are non-adjacent. Note that $y \in X^m_2$, hence by assumption and symmetry, we may assume that $v \notin B^m$. Therefore $v \in X^m$. Since X^m_1 is the union of components of X^m, and $y \in X^m_2$ is adjacent to v, it follows that $v \in X^m \setminus X^m_1$, and consequently $v \in X^m_2$. If $u \notin B^m$, then u-y-v is a P_3 in $G|X^m$, contrary to the first statement. Hence $u \in B^m$ and then u is mixed on the edge vy of $G|X^m_2$, a contradiction. This completes the proof. □

(5) $X^1_{12} \cup X^1_{23} \cup X^1_{13} \subseteq B^2 \cup S^2$.

Proof. Suppose that there exists $x' \in X^1_{ij} \setminus (B^2 \cup S^2)$ for some $1 \leq i \leq j \leq 3$; then $|L_2(x')| = 3$. Let $x_k \in X^1_{ij}$ and $a_k, b_k \in N(x_k) \cap B^1_{ij}$ be the vertices chosen to be precolored in the step creating L_2. Then x' is non-adjacent to $\{x_k, a_k, b_k\}$. The minimality of $|N(x_k) \cap B^1_{ij}|$ implies that there exist $a', b' \in (N(x') \cup B^1_{ij}) \setminus N(x_k)$. Since G is $2P_3$-free, there exists an edge between $\{a_k, b_k, x_k\}$ and $\{a', b', x'\}$. Specifically, there exists an edge between $\{a_k, b_k\}$ and $\{a', b'\}$. We may assume that $L_2(a_k) = \{i\}$ and a_k is adjacent to at least one of a', b'. Recall that L_1 is obtained by precoloring $\bigcup_{i=1}^{3} \{a_i, b_i, x_i\}$ and updating three times. It follows that $j \notin L_2(x')$, a contradiction. □

(6) *No vertex of B^3 is mixed on an edge of X_2.*

Proof. Suppose that there exists a path b'-x'_1-x'_2 such that $b' \in B^3_{ij}$ and $x'_1, x'_2 \in X^3_2$. Note that $x'_1, x'_2 \in X^2_2$ since L_3 is a subsystem of L_2. By Claim 4.1, X^2 is P_3-free. Hence $b' \in B^2_{ij}$. By Claim 4.3, there exists $b \in B^2_{ij}$ such that $b - x - y$ is a path where $x, y \in X^2_2$. Then in step 3, $\hat{B}_k \neq \emptyset$ and let $b \in \hat{B}_k$. By the construction of L_3 and since $x'_1, x'_2 \in X^3_2$, b is anticomplete to $\{b', x'_1, x'_2\}$. By the construction of $\hat{B}_k$, there exist $x_1, x_2 \in X^2_2$ such that b-x_1-x_2 is a path and b' is not mixed on x_1x_2. If $\{x_1, x_2\}$ is not anticomplete to $\{x'_1, x'_2\}$, then by Claim 4.1 $G|\{x_1, x_2, x'_1, x'_2\}$ is a K_4, a contradiction to the fact that $|V(G)| \geq 5$. Hence $\{x_1, x_2\}$ is anticomplete to $\{x'_1, x'_2\}$. Since G

is $2P_3$-free, there exists an edge between b' and $\{x_1, x_2\}$. Consequently, b' is complete to $\{x_1, x_2\}$. Now x_1 has two neighbors in B^2_{ij}, namely b and b'. By Claim 5, $x_1 \notin B^1_{ij}$. It follows that either $b \in X^1$ or $b' \in X^1$. If $b \in X^1$, then $b - x - y$ is a P_3 in X^1, contrary to Claim 4.1. Hence $b' \in X^1$. It follows that b'-x'_1-x'_2 is a P_3 in X^1, again contrary to Claim 4.1. This completes the proof. □

We are now ready to prove that it suffices to show that (G, L_5) induces a mininal list-obstruction of bounded size. Let C_t be an i-wide component of X^4_1 with $C_t = \{x_t, y_t\}$, and let a_t, b_t be the i-anchors of C_t that were chosen in step 5. By the definition of i-anchors, $L_4(a_t) \cap L_4(b_t) = \{i\}$ and $\{a_t, b_t\}$ is complete to C_t; therefore $c(a_t) = c(b_t) = i$ for every coloring c of $(G|\{x_t, y_t, a_t, b_t\}, L_4)$. Hence we can apply Claim 1 to L_4. By Claim 1 and Lemma 4.3.4, it is enough to show that (G, L_5) induces a bounded size list-obstruction.

(7) *X^5_2 is stable.*

Proof. Since $|V(G)| \geq 5$ and since no vertex of B^5 is mixed on an edge of $G|X^5_2$, by Claim 4.1 every component of X^5_2 has size at most 2. We may assume some component C of X^5_2 has size exactly 2, for otherwise the claim holds. Then C is a component of X^4_2. By Claim 4 and Claim 5, no vertex of X^4_2 has two neighbors in B^4_{ij}. Since every vertex in G has degree at least 3, every vertex of C has a neighbor in at least two of $B^4_{12}, B^4_{23}, B^4_{13}$. It follows that C is i-wide for some i and therefore $C \subseteq S^5 \cup B^5$, a contradiction. □

By Claim 4 and Claim 6, no vertex of X^5_2 has two neighbors in B^5_{ij}. Since every vertex in G has degree at least 3, it follows that every vertex of X^5_2 has exactly one neighbor in each of B^5_{ij}. Let Y_0, $Y_1, \ldots, Y_6$ be a partition of X^5_2 as follows. Let $x \in X^5_2$ and $a_k = N(x) \cap B^5_{ij}$ for $\{i, j, k\} = \{1, 2, 3\}$. If $\{a_1, a_2, a_3\}$ is a stable set, then $x \in Y_0$; if $E(G|\{a_1, a_2, a_3\}) = \{a_i a_j\}$, then $x \in Y_k$; and if $E(G|\{a_1, a_2, a_3\}) = \{a_i a_j, a_i a_k\}$, then $x \in Y_{i+3}$. Note that $G|\{a_1, a_2, a_3\}$ cannot be a clique since $V(G) \geq 5$. For each non-empty Y_s, pick $x_s \in Y_s$, and let $a_{sk} \in N(x_s) \cap B^5_{ij}$ for $\{i, j, k\} = \{1, 2, 3\}$. Let L_6 be the list system obtained by precoloring $\bigcup_{i=0}^{6} \{x_i, a_{i1}, a_{i2}, a_{i3}\}$ with c and updating three times.

(8) *For every $x \in X^5_2$, $|L_6(x)| \leq 2$*

Proof. Suppose there exists $y \in Y_i$ such that $|L_6(y)| = 3$; let $b_1 = N(y) \cap B^5_{23}$, $b_2 = N(y) \cap B^5_{13}$, and $b_3 = N(y) \cap B^5_{12}$. Then $\{a_{i1}, a_{i2}, a_{i3}, x_i\}$ and $\{b_1, b_2, b_3, y\}$ are disjoint sets. Note that

$c(a_{i1}), c(a_{i2}), c(a_{i3})$ can not all be pairwise different, and so by symmetry we may assume that $c(a_{i1}) = c(a_{i2}) = 3$ and $c(a_{i3}) = 2$. Thus, $a_{i1}a_{i2}$ is a non-edge. By the construction of L_6 and since $|L_6(y)| = 3$, the only possible edges between the sets $\{a_{i1}, a_{i2}, a_{i3}\}$ and $\{b_1, b_2, b_3\}$ are $a_{i3}b_2$, $a_{i1}b_3$ and $a_{i2}b_3$. Recall that every vertex of X_2^5 has exactly three neighbors in B^5, and so $N(y) \cap B^5 = \{b_1, b_2, b_3\}$ and $N(x_i) \cap B^5 = \{a_{i1}, a_{i2}, a_{i3}\}$. Since $G|\{a_{i1}, x_i, a_{i2}, b_1, y, b_2\}$ is not a $2P_3$, it follows that b_1 is adjacent to b_2. But this contradicts to the fact that both x_i and y belong to Y_i. □

Let (X_6^1, X_6^2, B^6, S^6) be the partition with respect to L_6. For every component $C_s \subseteq X_1^6$, let $\{i, j, k\} = \{1, 2, 3\}$ be such that there exists $x_s^k \in C_s$ with two adjacent neighbors in B_{ij}^6. Define $L_6'(x_k^s) = \{k\}$; let P be the set of all such vertices x_k^s, and let $L_6'(v) = L_6(v)$ for every $v \notin P$. Let L^* be the list system obtained from L_6' by updating with respect to P three times. Pick $x \in P$, then there exist $i, j \in \{1, 2, 3\}$ for which some $a, b \in N(x) \cap B_{ij}^6$ are adjacent. Then $L_6(a) = L_6(b) = \{i, j\}$. As a result, for every coloring c of $(G|\{x, a, b\}, L_6)$, $c(x) = k$. This implies that we can apply Claim 1 to L_6. By Lemma 4.3.4 and Claim 1, it is enough to prove that (G, L^*) induces a bounded size list-obstruction. Let (X_1^*, X_2^*, B^*, S^*) be the partition with respect to L^*. Then by Claim 4.1 X_1^*, X_2^* are empty. Now (G, L^*) satisfies the hypotheses of Lemma 4.4.1, and this finishes the proof of Lemma 4.4.2.

4.5 $P_4 + kP_1$-free minimal list-obstructions

In this section we prove that there are only finitely many $P_4 + kP_1$-free minimal list-obstructions. This also implies that there are only finitely many $P_4 + kP_1$-free 4-vertex-critical graphs.

4.5.1. *Let (G, L) be a minimal list-obstruction such that each list has at most two entries. Moreover, let G be $(P_4 + kP_1)$-free, for some $k \in \mathbb{N}$. Then $V(G)$ is bounded from above by a constant depending only on k.*

Proof. By Lemma 4.2.1, it suffices to prove that every propagation path in (G, L) has a bounded number of vertices. To see this, let $P = v_1\text{-}\ldots\text{-}v_n$ be a propagation path in (G, L) starting with color α, say. Consider v_1 to be colored with α, and update along P until every vertex is colored. Call this coloring c. Suppose that $n \geq 100k^2 + 100$. Our aim is to show that this assumption is contradictory. Recall condition (4.1) from the definition of propagation path: every edge v_iv_j with

$3 \leq i < j \leq n$ and $i \leq j-2$ is such that

$$S(v_i) = \alpha\beta \text{ and } S(v_j) = \beta\gamma,$$

where $\{1,2,3\} = \{\alpha, \beta, \gamma\}$.

First we suppose that there is a sequence $v_i, v_{i+1}, \ldots, v_j$ with $2 \leq i \leq j \leq n$ and $j - i \geq 5 + 2k$ such that $c(v_{i'}) = c(v_{i'+2})$ for all i' with $i \leq i' \leq j-2$. But then (4.1) implies that v_{i+1}-v_{i+2}-$\ldots$-v_j is an induced path, and thus G is not $P_4 + kP_1$-free, a contradiction.

Suppose now that there is an index i with $2 \leq i \leq \lceil n/2 \rceil - 3$ such that $c(v_i) = c(v_{i+2}) = \alpha$ and $c(v_{i+1}) = c(v_{i+3}) = \beta$. In particular, $L(v_{i+3}) = \{\alpha, \beta\}$. Now condition (4.1) of the definition of a propagation path implies that there cannot be an edge between v_i and v_{i+3}, and so v_i-v_{i+1}-v_{i+2}-v_{i+3} is an induced P_4. Therefore no such sequence exists.

We now pick k disjoint intervals of the form $\{j, \ldots, j+7+2k\} \subseteq \{\lceil n/2 \rceil + 1, \ldots, n\}$. As shown above, each of these intervals contains an index i' in its interior with $c(v_{i'}) = \alpha$. These $v_{i'}$ form a stable set and (4.1) implies that the induced path v_i-v_{i+1}-v_{i+2}-v_{i+3} is anticomplete to each $v_{i'}$, a contradiction to the fact that G is $P_4 + kP_1$-free.

Now suppose that there is an index i with $r+1 \leq i \leq \lceil (r+s)/2 \rceil - 3$ such that $c(v_i) = c(v_{i+2}) = \alpha$ and $c(v_{i+1}) = \beta$. From what we have shown above we know that $c(v_{i-1}) = c(v_{i+3}) = \gamma$, where $\{\alpha, \beta, \gamma\} = \{1,2,3\}$. Thus, we have $S(v_i) = \alpha\gamma$, $S(v_{i+1}) = \beta\alpha$, $S(v_{i+2}) = \alpha\beta$, and $S(v_{i+3}) = \gamma\alpha$. According to (4.1), the path v_i-v_{i+1}-v_{i+2}-v_{i+3} is induced.

Pick a vertex v_j with $\lceil n/2 \rceil + 1 \leq j \leq n$. According to (4.1), v_j is anticomplete to the path v_i-v_{i+1}-v_{i+2}-v_{i+3} unless one of the following holds.

(a) $S(v_j) = \alpha\beta$,

(b) $S(v_j) = \alpha\gamma$,

(c) $S(v_j) = \beta\gamma$, or

(d) $S(v_j) = \gamma\beta$.

Let us say that v_j is of *type A* if it satisfies one of the above conditions. If v_j is not of type A, we say it is of *type B*.

We claim that there are at most $3k-3$ vertices of type B. To see this, suppose there are at least $3k-2$ vertices of type B. By definition, each vertex of type B is anticomplete to the set the path

v_i-v_{i+1}-v_{i+2}-v_{i+3}. Since (G, L) is a minimal obstruction and not every vertex is of type B, the graph induced by the vertices of type B is 3-colorable. Picking the vertices of the majority color yields a set S of k independent vertices of type B. But now the set $\{v_i, \ldots, v_{i+3}\} \cup S$ induces a $P_4 + kP_1$ in G, a contradiction.

So, there are at most $3k-3$ vertices of type B. Suppose there are more than $(3k-2)(7+2k)$ many vertices of type A. Then there is an index $t \geq \lceil n/2 \rceil + 1$ such that $v_t + j'$ is of type A for all $j' \in \{0, \ldots, 6+2k\}$. Suppose that there is an index $j' \in \{0, \ldots, 5+2k\}$ such that $c(v_{t+j'}) = \alpha$. Then $S(v_{t+j'+1}) = \cdot\, \alpha$, in contradiction to the fact $v_{t+j'+1}$ is of Type A. So, for all $j' \in \{0, \ldots, 5+2k\}$ we have that $c(v_{t+j'}) \neq \alpha$, in contradiction to what we have shown above. Summing up, n is bounded by $2(3k-2)(7+2k)+1$ if there is an index i with $2 \leq i \leq \lceil n/2 \rceil - 3$ such that $c(v_i) = c(v_{i+2}) = \alpha$ and $c(v_{i+1}) = \beta$.

Hence, our assumption $n \geq 100k^2 + 100$ implies that $c(v_i) \neq c(v_{i+2})$ for all i with $2 \leq i \leq \lceil n/2 \rceil - 3$. This means that, without loss of generality,

$$c(v_i) = \begin{cases} 1, & i = 1 \ (3) \\ 2, & i = 2 \ (3) \\ 3, & i = 0 \ (3) \end{cases} \tag{4.12}$$

for all i with $2 \leq i \leq \lceil n/2 \rceil - 3$.

Consider the path v_4-v_5-$\ldots$-v_{7+2k}. Since G is $P_4 + kP_1$-free, this is not an induced path. Hence, there is an edge of the form $v_i v_j$ with $i < j$. If $S(v_i) = \alpha\beta$, we must have $S(v_j) = \beta\gamma$, due to (4.1). Consequently, $S(v_{i-1}) = \beta\gamma$, and $S(v_{j+1}) = \alpha\beta$. In particular, (4.1) implies that v_{i-1} is non-adjacent to v_{j+1}, and so v_{i-1}-v_i-v_j-v_{j+1} is an induced path.

Like above, we now pick k disjoint intervals of the form $\{j, \ldots, j+7+2k\} \subseteq \{\lceil n/2 \rceil + 1, \ldots, n\}$. Each of these intervals contains an index i' in it's interior with $c(v_{i'}) = \alpha$. These $v_{i'}$ form a stable set and (4.1) implies that the induced path v_{i-1}-v_i-v_j-v_{j+1} is anticomplete to each $v_{i'}$, a contradiction. This completes the proof. □

Using the above statement, we can now derive our main lemma.

4.5.2. *There are only finitely many $P_4 + kP_1$-free minimal list-obstructions, for all $k \in \mathbb{N}$.*

Proof. Let (G, L) be a $P_4 + kP_1$-free minimal list-obstruction. If G is P_4-free, we are done, since there is only a finite number of P_6-free minimal obstructions. So, we may assume that G contains

an induced P_4, say v_1-v_2-v_3-v_4. Let $R = V(G) \setminus N(\{v_1, v_2, v_3, v_4\})$. Let S be a maximal stable set in R; then every vertex of $V(R) \setminus S$ has a neighbor in S. Since G is $P_4 + kP_1$-free, it follows that $|S| \leq k-1$, and so $\{v_1, v_2, v_3, v_4\} \cup S$ is a dominating set of size at most $k+3$ in G. Now Lemma 4.5.2 follows from Lemma 4.3.5 and Lemma 4.5.1. □

4.6 Necessity

The aim of this section is to prove the following two statements.

4.6.1. *There are infinitely many H-free 4-vertex-critical graphs if H is a claw, a cycle, or $2P_2 + P_1$.*

Here, a *claw* is the graph consisting of a central vertex plus three pairwise non-adjacent pendant vertices attached to it. In the list-case, the following variant of this statement holds.

4.6.2. *There are infinitely many H-free minimal list-obstructions if H is a claw, a cycle, $2P_2 + P_1$, or $2P_3$.*

We remark that Lemma 4.6.1 implies the following. Whenever H is a graph containing a claw, a cycle, or $2P_2 + P_1$ as an induced subgraph, there are infinitely many H-free 4-vertex-critical graphs. A similar statement is true with respect to Lemma 4.6.2 and minimal list-obstructions.

4.6.1 Proof of Lemma 4.6.1

Recall that there are infinitely many 4-vertex-critical claw-free graphs. For example, this follows from the existence of 4-regular bipartite graphs of arbitrarily large girth (cf. [38] for an explicit construction of these) whose line graphs are necessarily 4-chromatic. Moreover, there are 4-chromatic graphs of arbitrarily large girth, which follows from a classical result of Erdős [18]. This, in turn, implies that there exist 4-vertex-critical graphs of arbitrary large girth. Putting these two remarks together, we see that if H is the claw or a cycle, then there are infinitely many 4-vertex-critical graphs.

We now recall a construction due to Pokrovskiy [47] which gives an infinite family of 4-vertex-critical P_7-free graphs. It is presented in more detail in our earlier work [10].

For each $r \geq 1$, let G_r be the graph defined on the vertex set $v_0, \ldots, v_{3r}$ with edges as follows. For all $i \in \{0, 1, \ldots, 3r\}$ and $j \in \{0, 1, \ldots, r-1\}$, the vertex v_i is adjacent to v_{i-1}, v_{i+1}, and v_{i+3j+2}. Here, we consider the indices to be taken modulo $3r+1$. The graph G_5 is shown in Figure 4.1.

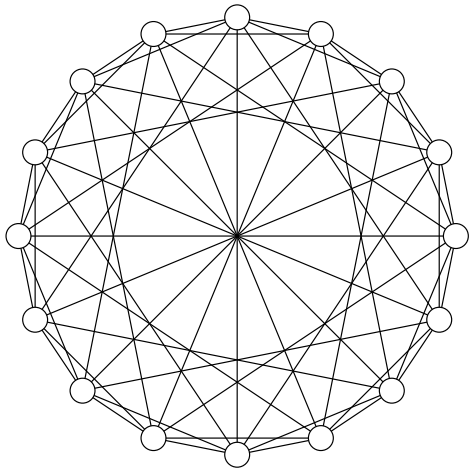

Figure 4.1: A circular drawing of G_5

Up to permuting the colors, there is exactly one 3-coloring of $G_r \setminus v_0$. Indeed, we may assume that v_i receives color i, for $i = 1, 2, 3$, since $\{v_1, v_2, v_3\}$ forms a triangle in G_r. Similarly, v_4 receives color 1, v_5 receives color 2 and so on. Finally, v_{3r} receives color 3. It follows that G_r is not 3-colorable, since v_0 is adjacent to all of v_1, v_2, v_{3r}.

As the choice of v_0 was arbitrary, we know that G_r is 4-vertex-critical. The graph G_r is $2P_2+P_1$-free which can be seen as follows.

(1) *For all r the graph G_r is $2P_2 + P_1$-free.*

Proof. Suppose there is some r such that G_r is not $2P_2 + P_1$-free. Let $v_{i_1}, \ldots, v_{i_5}$ be such that $G_r[\{v_{i_1}, \ldots, v_{i_5}\}]$ is a $2P_2 + P_1$. Since G_r is vertex-transitive, we may assume that $i_1 = 1$ and $N(v_{i_1}) \cap \{v_{i_2}, \ldots, v_{i_5}\} = \emptyset$. In particular, $i_2, \ldots, i_5 \neq 0$.

Consider $G_r \setminus v_0$ to be colored by the coloring c proposed above, where each v_i receives the color $i \mod 3$. Due to the definition of G_r, v_{i_1} is adjacent to every vertex of color 3, and thus $c(v_j) \neq 3$ for all $j \in \{i_2, \ldots, i_5\}$.

We may assume that $c(v_{i_2}) = c(v_{i_4}) = 1$, $c(v_{i_3}) = c(v_{i_5}) = 2$, and both $v_{i_2}v_{i_3}$ and $v_{i_4}v_{i_5}$ are edges of $E(G_r)$. For symmetry, we may further assume that $i_2 < i_4$. Due to the definition of G_r, v_{i_2} and v_{i_4} are adjacent to every vertex of color 2 with a smaller index, and thus $i_4 < i_3$. But now $i_2 < i_4 < i_3$, a contradiction to the fact that $v_{i_2}v_{i_3} \in E(G_r)$. This completes the proof. □

Consequently, there are infinitely many $2P_2 + P_1$-free 4-vertex-critical graphs, as desired.

4.6.2 Proof of Lemma 4.6.2

In view of Lemma 4.6.1, it remains to prove that there are infinitely many $2P_3$-free minimal list-obstructions.

For all $r \in \mathbb{N}$, let H_r be the graph defined as follows. The vertex set of H_r is $V(H_r) = \{v_i : 1 \leq i \leq 3r-1\}$. There is an edge from v_1 to v_2, from v_2 to v_3 and so on. Thus, $P := v_1$-v_2-$\ldots$-v_{3r-1} is a path. Moreover, there is an edge between a vertex v_i and a vertex v_j if $i \leq j-2$, $i \equiv 2 \mod 3$, and $j \equiv 1 \mod 3$. There are no further edges. The graph H_5 is shown in Figure 4.2.

The list system L is defined by $L(v_1) = L(v_{3r-1}) = \{1\}$ and, assuming $2 \leq i \leq 3r-2$,

$$L(v_i) = \begin{cases} \{2,3\}, & \text{if } i \equiv 0 \mod 3 \\ \{1,3\}, & \text{if } i \equiv 1 \mod 3 \\ \{1,2\}, & \text{if } i \equiv 2 \mod 3 \end{cases}.$$

Next we show that the above construction has the desired properties.

(2) *The pair (H_r, L) is a minimal $2P_3$-free list-obstruction for all r.*

Proof. Let us first show that, for any r, H_r is not colorable. Consider the partial coloring c that assigns color 1 to v_1. Since $L(v_2) = \{1,2\}$, the coloring can be updated from v_1 to v_2 by putting $c(v_2) = 2$. Now we can update the coloring from v_2 to v_3 by putting $c(v_3) = 3$. Like this we update the coloring along P until v_{3r-2} is colored. However, we have to put $c(v_{3r-2}) = 1$, in contradiction to the fact that $L(v_{3r-1}) = \{1\}$. Thus, H_r is not colorable.

Next we verify that (H_r, L) is a minimal list-obstruction. If we delete v_1 or v_{3r-1}, the graph becomes colorable. So let us delete a vertex v_i with $2 \leq i \leq 3r-2$. We can color $(H_r \setminus v_i, L)$ as follows. Give color 1 to v_1 and update along P up to v_{i-1}. Moreover, give color 1 to v_{3r-1} and update along P backwards up to v_{i+1}. Call this coloring c.

To check that c is indeed a coloring, we may focus on the non-path edges for obvious reasons. Pick an edge between a vertex v_j and a vertex v_k with $j \leq k-2$, if any. By definition, $j \equiv 2 \mod 3$ and $k \equiv 1 \mod 3$. If $j < i < k$, $c(v_j) = 2$ and $c(v_k) = 3$. Moreover, if $j < k < i$, $c(v_j) = 2$ and $c(v_k) = 1$. Finally, if $i < j < k$, $c(v_j) = 1$ and $c(v_k) = 3$. So, c is indeed a coloring of $H_r \setminus v_i$ and it remains to prove that H_r is $2P_3$-free.

Suppose this is false, and let r be minimum such that H_r contains an induced $2P_3$. Let F be a copy of such a $2P_3$ in H_r. It is clear that $r \geq 2$. Note that $H_r \setminus N(v_2)$ is the disjoint union of

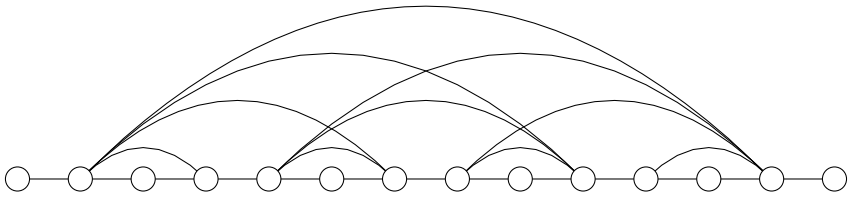

Figure 4.2: A drawing of H_5. The vertices v_1 to v_{14} are shown from left to right.

complete graphs of order 1 and 2, and so $v_2 \notin V(F)$. Since $N(v_1) = \{v_2\}$, we know that $v_1 \notin V(F)$. Moreover, as $F \setminus (N(v_5) \cup \{v_1, v_2\})$ is the disjoint union of complete graphs of order 1 and 2, we deduce that $v_5 \notin V(F)$. But $F' := F \setminus \{v_1, v_2, v_3\}$ is isomorphic to H_{r-1}, and thus the choice of r implies that F' is $2P_3$-free. Consequently, $v_3 \in V(F)$. Since $N(v_3) = \{v_2, v_4\}$ and $v_2 \notin V(F)$, we know that $v_4 \in V(F)$. Finally, the fact that $N(v_4) = \{v_2, v_3, v_5\}$ implies that v_3 and v_4 both have degree one in F, and they are adjacent, a contradiction. □

4.7 Proof of Theorem 4.1.2 and Theorem 4.1.3

We now prove our main results. We start with a lemma.

4.7.1. *For every graph H, one of the following holds.*

1. *H contains a cycle, a claw or $2P_2 + P_1$.*
2. *$H = 2P_3$.*
3. *H is contained in P_6.*
4. *There exists $k > 1$ such that H is contained in $P_4 + kP_1$.*

Proof. We may assume that H does not contain $2P_2 + P_1$, a cycle, or a claw. It follows that every component of H induces a path. Let $H_1, H_2, \ldots H_k$ be the components of H, ordered so that $|H_1| \geq |H_2| \geq \ldots \geq |H_k|$.

If $|H_2| \geq 2$, then, since H is $2P_2 + P_1$-free, it follows that $k = 2$, $|H_1| \leq 3$, and $|H_2| \leq 3$, and so either H is contained in P_6 or $H = 2P_3$. This proves that $|H_2| = \ldots = |H_k| = 1$.

If $|H_1| \geq 5$, then since H is $2P_2 + P_1$-free, it follows that $k = 1$, and H is contained in P_6. This proves that $|H_1| \leq 4$, and so H is contained in $P_4 + (k-1)P_1$. This proves Lemma 4.7.1. □

Next we prove Theorem 4.1.2, which we restate:

4.1.2. *Let H be a graph. There are only finitely many H-free 4-vertex-critical graphs if and only if H is an induced subgraph of P_6, $2P_3$, or $P_4 + kP_1$ for some $k \in \mathbb{N}$.*

Proof. If H contains a cycle, a claw or $2P_2 + P_1$, then there is an infinite list of 4-vertex-critical graphs by Lemma 4.6.1. By Lemma 4.7.1, $H = 2P_3$, H is contained in P_6, or for some $k > 1$, H is contained in $P_4 + kP_1$, and Lemmas 4.4.2, 4.3.2 and 4.5.2, respectively, imply that there are there are only finitely many H-free 4-vertex-critical graphs. □

Finally, we prove the list version of the result, Theorem 4.1.3, which we restate:

4.1.3. *Let H be a graph. There are only finitely many H-free minimal list-obstructions if and only if H is an induced subgraph of P_6, or of $P_4 + kP_1$ for some $k \in \mathbb{N}$.*

Proof. If H contains a cycle, a claw, $2P_2 + P_1$ or $2P_3$, then there is an infinite list of obstructions by Lemma 4.6.2. By Lemma 4.7.1, H is contained in P_6, or for some $k > 1$, H is contained in $P_4 + kP_1$. Now Lemmas 4.3.2 and 4.5.2, respectively, imply that there are there are only finitely many H-free list-obstructions. □

Chapter 5

Scheduling When You Don't Know the Number of Machines

5.1 Introduction

In scheduling problems, there are many ways to model uncertainty in the jobs, including online algorithms [1; 2], stochastic scheduling [45], and work on schedules that are good against multiple objective functions [4; 43; 51]. But there is much less work studying the possibility of uncertainty in the machines. Motivated by the need to understand how to make scheduling decisions without knowing how many machines we will have, in this chapter, we consider the following model. We are given a set of n jobs, J, with a known processing times $p(j)$ for each job j, and a number M, which is an upper bound on the number of machines we might have. An algorithm must commit, before knowing how many machines there are, to grouping the jobs into M *bags*, where each job is assigned to exactly one of the bags. We call this step the *packing* step. Only after completing the packing step do we learn the number of machines m. We now need to compute a schedule, with the restriction that we must keep the bags together, that is, we will assign one or more bags to each machine. We call this step the *scheduling* step. As in other robust problems, we want to do well against all possible numbers of machines. We therefore evaluate our schedule by the ratio of the makespan of our schedule, $ALG(m, M)$, to the makespan of a schedule that knew m in advance, opt_m, taking the worst case over all possible values of m. If an algorithm always provides a ratio

of at most α, where $\alpha = \max_{1 \leq m \leq M} \frac{ALG(m,M)}{\text{opt}_m}$, we call it α-robust. (We may also consider scenarios in which there are different upper and lower bounds on the range of m; the definition of robustness extends in the obvious way.)

Our main result is an algorithm for minimizing makespan on parallel machines. which is $(\frac{5}{3}+\epsilon)$-robust; and we show a lower bound of 4/3 on the robustness of any algorithm for minimizing makespan on parallel machines. As with many scheduling problems, there are two different aspects to address. One is the load-balancing aspect, but in this two-stage problem, it seems that one wants to create bags of a variety of different sizes, in order to allow a more balanced final schedule. The second is to deal with large jobs, and to handle cases where one or several large jobs are the dominant term in the makespan. Large jobs seem to provide a particular challenge in this problem, and much of our algorithm and analysis are devoted to handling various cases involving large jobs.

In order to focus on the load balancing issues, we also consider the "continuous" case where we have a set of infinitesimal jobs. That is, in the packing stage, we simply need to divide our total load into M bags. In traditional makespan scheduling, this case is trivial, we would just divide the load into M equal pieces and achieve an optimal makespan. But in this two stage-problem, even the continuous case is challenging. We can, however, obtain significantly stronger results than in the discrete case, showing an upper bound of 1.233 and a lower bound of 1.207 on the robustness.

The continuous case also models a problem in *fair allocation*. In fair allocation, you typically have resources that you want to split "fairly" among several parties. The literature on this problem is vast and we will not attempt to summarize it here. We will only observe that we are solving a problem in fair allocation that has not previously been studied, to our knowledge. We are given some objects to share, and everyone agrees on the values, but we don't know how many people will be sharing them. We place the objects into bags, and have the restriction that each person must take a subset of the bags. In the makespan variant, we are minimizing the maximum amount that anyone gets. Motivated by fairness, we also consider a version where you want to minimize the difference between the maximum allocation and the minimum allocation (this objective makes sense in fair allocation, but not necessarily in scheduling). Here we consider a case where we know a lower bound of αM on the eventual number of machines, and can show a lower bound of $\min\{2/3, 2/(4\alpha+1)\}A$ on the difference and we can obtain an upper bound of $\min\{2/3, 1/(\alpha+1)\}A$, where A is the average load.

5.1.1 Overview of this chapter and a Lower Bound

We give a brief overview of this chapter, and for intuition, a simple lower bound.

In Section 5.2, we consider the case when all jobs are infinitesimal. We give an algorithm which is approximately 1.233 robust. And we also show a lower bound which is approximately 1.207. For the infinitesimal case, we also consider the objective of minimizing the maximum difference between the most loaded and least loaded machine. For this case, and the average load is A, if we know that the eventual number of machines is in the range $[\alpha M, M]$, we can show an asymptotic lower bound of $\min\{2/3, 2/(4\alpha+1)\}A$ on the difference and we can obtain an upper bound of $\min\{2/3, 1/(\alpha+1)\}A$.

In Section 5.3, we consider the general case with arbitrarily sized jobs. We give an algorithm which gives a robust ratio of $\frac{5}{3}+\epsilon$, breaking the simple 2 bound that can be obtained by running LPT on M sets and then repeatedly merging the two smallest sets until m sets remain. In our algorithm, we first calculate the optimal schedule for $m \in \{\frac{M}{2}, \frac{3M}{4}, M\}$ within a factor of $1+\epsilon$ using the algorithm in [34]. We then take one of these schedules and partition the jobs that were scheduled on some machines into a larger number of sets, which we call *bags*. After learning how many machines we have, we place the bags on the machines. This algorithm is more involved than the previous ones, and has several cases, based on the values of opt_M, $\text{opt}_{3M/4}$ and $\text{opt}_{M/2}$ and demonstrates how a more careful investigation of the packing and scheduling steps can lead to improved bounds.

We conclude the introduction with a simple lower bound. Consider the following example. Let the upper bound on the number of machines, M, be 3 and assume we have $n = 6$ identical jobs, each of which has processing time 1. We will prepare 3 bags which will ultimately have to be scheduled on either 1, 2 or 3 machines. Note that the unconstrained optimal makespan for 1, 2 and 3 machines are 6, 3 and 2 respectively. First, consider the packing which gives each bag 2 jobs. Then the makespans are 6, 4, and 2 respectively if there are 1, 2 and 3 machines. Hence this algorithm is $\max\{6/6, 4/3, 2/2\} = 4/3$-robust. Next consider the packing which places $\{1, 2, 3\}$ jobs in each bag respectively. Then the makespans are 6, 3, and 3 respectively if there are 1, 2 and 3 machines, which makes this algorithm $\max\{6/6, 3/3, 3/2\} = 3/2$-robust. In this example, the former algorithm is better. Moreover, this example demonstrates that 4/3 is a lower bound on the robustness of any algorithm. Note that there exists a same lower bound for any number of M: Consider an arbitrary

M and $n = 2M$ identical jobs with each processing time 1, if we put at least one bag with at least 3 jobs, then the robust ratio is at least 3/2; otherwise we put 2 jobs in each bag and it provides a robust ratio which is at least 4/3.

5.2 Scheduling infinitesimal jobs

Throughout this paper, we use $p(j)$ to denote the processing time of job j. For any set of jobs S, we use $p(S) = \sum_{p_i \in S} p(i)$ to denote the sum of the processing times of jobs in S. We informally say a job set S is *big* if the value of $p(S)$ is large and *small* otherwise.

We now consider the case of infinitesimal jobs. Suppose we are given a job set J with all infinitesimal jobs such that $p(J) = s$, for some $s > 0$. We first pack the jobs to $M \geq 3$ sets and then schedule the bags on m machines, where $m \in [1, M]$ is only known after we pack the jobs. Let $ALG(m, M)$ be the makespan of scheduling the bags on m machines, then our objective is to minimize the robust ratio, $\alpha = \max_m \left\{ \frac{ALG(m,M)}{\text{opt}_m} \right\}$. Recall opt_m is the makespan of a schedule that knew m in advance, and specifically $\text{opt}_m = s/m$ when all jobs are infinitesimal.

The main idea in the packing is to produce a set of bags with a diverse set of sizes. More precisely, we consider the following packing, which we call packing PC. $S_1, S_2, \ldots, S_M$ are the bags: for $i = 1, 2, 3, \ldots, 2\lfloor M/3 \rfloor$, $p(S_i) = \frac{ks}{M - \lceil \frac{i}{2} \rceil} - \frac{ks}{2(M-1)}$; for $j = 2\lfloor M/3 \rfloor + 1, 2\lfloor M/3 \rfloor + 2, \ldots, M$, $p(S_j) = \frac{ks}{M}$, were k is a parameter which only depends on M, chosen to ensure that $\sum_{i=1}^{M} p(S_i) = s$. Specifically, $k = 1/(2 \cdot \sum_{i=1}^{\lfloor M/3 \rfloor} \frac{1}{M-i} - \left\lfloor \frac{M}{3} \right\rfloor \frac{1}{M(M-1)} + 3 \left\{ \frac{M}{3} \right\} \frac{1}{M}) \approx 1.233$ when M is large (Here we use $\{\}$ to denote the integer remainder). And we will show that for all M, $k \leq 1.2333$. See Figure 5.1 to see how k changes with M.

Using packing PC to put the jobs into bags, we obtain the following theorem.

5.2.1. *For $m \in [1, M]$, there exists a schedule with makespan at most $k\text{opt}_m \leq 1.2333\text{opt}_m$ which schedules $\{S_1, S_2, \ldots, S_M\}$ on m machines.*

Proof. We first consider the case when $m \geq M/2$, and we schedule the bags as follows. Let $t = M - m$. For machine $i = 1, 2, \ldots, t$, we schedule bags S_i and S_{2t-i+1} on machine i; for machine $j = t+1, \ldots, m$, we schedule bag S_{j+t} on machine i . The machines with one bag are all within the bound, since for $i \leq 2\left\lfloor \frac{M}{3} \right\rfloor$, $p(S_i) = \frac{ks}{M - \lceil \frac{i}{2} \rceil} - \frac{ks}{2(M-1)} \leq \frac{ks}{M - M/3} - \frac{ks}{2M} = \frac{ks}{M} \leq \frac{ks}{M} \leq$

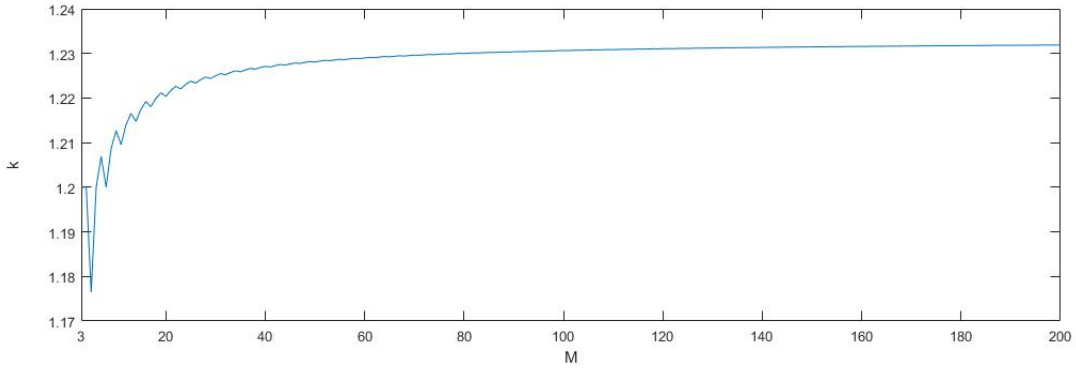

Figure 5.1: k as a function of M.

$\frac{ks}{m} = k\text{opt}_m$. Therefore it remains to bound the load on machines with two bags. We will use $L(i,t)$ to denote the load on the machine i for a particular value of t. We therefore need to prove that $L(i,t) = p(S_i) + p(S_{2t-i+1}) \leq k\text{opt}_m$ for $1 \leq i \leq t$, $t = M - m \leq \left\lfloor \frac{M}{2} \right\rfloor$. Observe that when i is even, $L(i,t) = p(S_i) + p(S_{2t-i+1}) = p(S_{i-1}) + p(S_{2t-i+2}) = L(i-1,t)$, hence we may assume that i is odd. Since $i \leq \left\lfloor \frac{M}{2} \right\rfloor \leq 2 \left\lfloor \frac{M}{3} \right\rfloor$, $p(S_i) = \frac{ks}{M - \frac{i+1}{2}} - \frac{ks}{2(M-1)}$.

We first consider the subcase when $2t - i + 1 \geq 2 \left\lfloor \frac{M}{3} \right\rfloor + 1$, then $p(S_{2t-i+1}) = \frac{ks}{M}$. Since i is odd, $i \leq 2t - 2 \left\lfloor \frac{M}{3} \right\rfloor - 1$. Hence we have

$$\begin{aligned} L(i,t) &= \frac{ks}{M - \frac{i+1}{2}} - \frac{ks}{2(M-1)} + \frac{ks}{M} \\ &\leq \frac{ks}{M - t + \lfloor M/3 \rfloor} + \frac{ks(M-2)}{2(M-1)M} \end{aligned} \tag{5.1}$$

Recall that $\text{opt}_m = \frac{ks}{m} = \frac{ks}{M-t}$. Since $t \geq \left\lfloor \frac{M}{3} \right\rfloor + \frac{i}{2}$, $t \geq \left\lfloor \frac{M}{3} \right\rfloor + 1$. Using (5.1), it follows

that

$$\begin{aligned}
& L(i,t) - \frac{ks}{M-t} \\
&= \frac{ks}{M+\lfloor M/3\rfloor - t} + \frac{ks(M-2)}{2(M-1)M} - \frac{ks}{M-t} \\
&\leq \frac{ks(M-2)}{2(M-1)M} - \frac{ks\lfloor M/3\rfloor}{(M+\lfloor M/3\rfloor - t)(M-t)} \\
&\leq \frac{ks(M-2)}{2(M-1)M} - \frac{ks\lfloor M/3\rfloor}{(M-1)(M-\lfloor M/3\rfloor - 1)} \\
&= ks\frac{M^2-3M+2-(3M-2)\lfloor M/3\rfloor}{2M(M-1)(M-\lfloor M/3\rfloor-1)} \\
&\leq ks\frac{M^2-3M+2-(3M-2)(M/3-\frac{2}{3})}{2M(M-1)(M-\lfloor M/3\rfloor-1)} \\
&= ks\frac{-M/3+\frac{2}{3}}{2M(M-1)(M-\lfloor M/3\rfloor-1)} \ < \ 0\ .
\end{aligned}$$

That is, $L(i,t) \leq \dfrac{ks}{M-t} = k\text{opt}_m$. Next, we consider the subcase that that $2t-i+1 \leq 2\left\lfloor \dfrac{M}{3}\right\rfloor$. Now, we have $p(S_{2t-i+1}) = \dfrac{ks}{M-\frac{2t-i+1}{2}} - \dfrac{ks}{2(M-1)} = \dfrac{ks}{M+(i+1)/2-(t+1)} - \dfrac{ks}{2(M-1)}$ (Recall that we assume i is odd then $2t-i+1$ is even). Hence we have

$$\begin{aligned}
L(i,t) &= \frac{ks}{M-\frac{i+1}{2}} + \frac{ks}{M+\frac{i+1}{2}-(t+1)} - \frac{ks}{M-1} \\
&= \frac{ks(2M-t-1)}{(M-\frac{i+1}{2})(M+\frac{i+1}{2}-(t+1))} - \frac{ks}{M-1} \\
&= \frac{ks(2M-t-1)}{-\frac{1}{4}(i-t)^2+\frac{1}{4}(t+1)^2+M^2-(t+1)M} \\
&\quad - \frac{ks}{M-1} \\
&\leq \frac{ks(2M-t-1)}{-\frac{1}{4}(1-t)^2+\frac{1}{4}(t+1)^2+M^2-(t+1)M} \\
&\quad - \frac{ks}{M-1} \\
&= \frac{ks(2M-t-1)}{(M-1)(M-t)} - \frac{ks}{M-1} = \frac{ks}{M-t}\ .
\end{aligned}$$

The inequality holds because $1 \leq i \leq t$. This proves that for the case $m \geq M/2$, the makespan of our schedule is at most $\dfrac{ks}{M-t} = k\text{opt}_m$.

Next, we consider the case when $1 \leq m < M/2$. Let $x \geq 2$ be the integer such that $\frac{M}{x+1} \leq m < \frac{M}{x}$. Let $t = M - mx$. For machine $i = 1, 2, \ldots, t$, we schedule bags S_i, S_{2t-i+1},

$S_{i+2t}, \ldots, S_{i+xt}$ on such machine; for machine $j = t+1, \ldots, m$, we schedule bags $S_{j+xt}, S_{j+xt+(m-t)}$, $S_{j+xt+2(m-t)} \cdots, S_{j+xt+(x-1)(m-t)}$ on such machine. Recall that $\forall i \leq M$, $p(S_i) \leq \frac{ks}{M}$. Observe that we schedule x bags on the last $m-t$ machines, hence the processing times of jobs on such those machines are at most $\frac{xks}{M} = \frac{ks}{M/x} \leq \frac{ks}{m} = k\text{opt}_m$. The processing time of the bags scheduled on machine $i \leq t$ is (note that $m = (M-t)/x$):

$$\begin{aligned} p(S_i) + p(S_{2t-i+1}) + \sum_{j=2}^{x} p(S_{i+jt}) &= L(i,t) + \sum_{j=2}^{x} p(S_{i+jt}) \\ &\leq \frac{ks}{M-t} + (x-1)\frac{ks}{M} \\ &\leq \frac{kxs}{M-t} = k\text{opt}_m \end{aligned}$$

The last is to show a bound of k. Recall that k is chosen to ensure that $\sum_{i=1}^{M} p(S_i) = s$, so we have (here we use $\{\}$ to denote the integer remainder),

$$\begin{aligned} \sum_{i=1}^{M} p(S_i) &= 2 \cdot \sum_{i=1}^{\lfloor M/3 \rfloor} \left(\frac{ks}{M-i} - \frac{ks}{2(M-1)}\right) \\ &\quad + (M - 2\lfloor M/3 \rfloor) \cdot \frac{ks}{M} \\ &= 2 \cdot \sum_{i=1}^{\lfloor M/3 \rfloor} \frac{ks}{M-i} - \lfloor M/3 \rfloor \frac{ks}{M(M-1)} \\ &\quad + 3\{M/3\}\frac{ks}{M} \ . \end{aligned} \tag{5.2}$$

We can solve (5.2) for k and obtain that $k = 1/b(M)$ where $b(M) = 2 \cdot \sum_{i=1}^{\lfloor M/3 \rfloor} \frac{1}{M-i} - \left\lfloor \frac{M}{3} \right\rfloor \frac{1}{M(M-1)} + 3\{\frac{M}{3}\}\frac{1}{M}$. Note that when M is large,

$$\begin{aligned} b(M) &\approx 2 \sum_{i=M-\lfloor M/3 \rfloor}^{M-1} \frac{1}{i} \approx 2(\ln(M-1) - \ln(2M/3)) \\ &\approx 2 \ln 1.5 \end{aligned}$$

Hence $k = 1/b(M) \approx 1/(2 \ln 1.5) \approx 1.233$ when M is large. We verify by computer that when

$M < 10000$. $k \leq 1.2333$. And for $M \geq 10000$,

$$\begin{aligned} b(M) &\geq 2 \cdot \int_0^{\lfloor M/3 \rfloor} \frac{1}{M-i} - \frac{1}{3(M-1)} \\ &\geq 2ln\frac{M - \lfloor M/3 \rfloor}{M} - 4 \cdot 10^{-5} \geq 0.81089 \end{aligned}$$

So $k \leq 1/0.81089 \leq 1.2333$. □

We can also show a lower bound. That is, we can show that, no matter how you divide the jobs, you cannot achieve a robust ratio below 1.207, which is close to (but does not exactly match) our upper bound. We state the theorem in terms of a function $Q(M)$ which we define precisely below and which, for large M is 1.207.

5.2.2. *Let $S_1, S_2, \ldots, S_M$ be M bags of infinitesimal jobs such that $\sum_{i=1}^{M} p(S_i) = s$ for some $s > 0$. Assume there exists a constant k' such for all $m \in [1, M]$, we can schedule $\{S_1, S_2, \ldots, S_M\}$ on m machines with makespan at most $k' opt_m$. Then $k' \geq Q(M)$, where $Q(M) = \max_{t \leq \frac{M}{2}, t \in N} \frac{1}{\frac{t}{M-t} + \frac{M-2t}{M}} \approx$ 1.207.*

Proof. We may assume that $p(S_1) \leq p(S_2) \leq \cdots \leq p(S_M)$. We first prove the following statement.

$$\begin{aligned} &\textit{For } t \leq M/2\textit{, there exists a schedule of } \{S_1, S_2, \ldots, S_M\} \textit{ on } m = M - t \textit{ machines} \\ &\textit{with minimum makespan such that } S_1, S_2, \ldots, S_{2t} \textit{ are on the first } t \textit{ machines.} \end{aligned} \tag{5.3}$$

Let $\{T_1, T_2, \ldots, T_m\}$ be a schedule of $\{S_1, S_2, \ldots, S_M\}$ on m machines with minimum makespan, where T_i is a set of bags scheduled on machine i. We may assume that every T_i contains at least one bag. Since we schedule M bags on $M - t$ machines, there are at least $M - 2t$ machines contain exactly one bag. We rename the machines such that $T_{t+1}, T_{t+2}, \ldots, T_{t+(M-2t)} = T_m$ contain only one bag. Suppose that at least one of $S_1, S_2, \ldots, S_{2t}$ is not scheduled on the first t machines, that is, there exists $i \leq 2t$, $j \geq t+1$ such that $T_j = \{S_i\}$. Then since $|\bigcup_{i=1}^{t} T_i| = M - |\bigcup_{i=t+1}^{m} T_i| = 2t$, there exists $i' \geq 2t+1$, $j' \leq t$ such that $S_{i'} \in T_{j'}$. Define $T'_j = \{S_{i'}\}$, $T'_{j'} = T_{j'} \setminus \{S_{i'}\} \cup \{S_i\}$, and $T'_\ell = T_\ell$ for $\ell \neq j, j'$. Since $p(S_{i'}) \geq p(S_i)$, $p(T'_{j'}) \leq p(T_{j'})$. Note also that $p(T'_j) \leq \max_{i=1}^{M}\{p(S_i)\} \leq \max_{i=1}^{m}\{p(T_i)\}$. Hence $\max_{i=1}^{m}\{p(T'_i)\} \leq \max_{i=1}^{m}\{p(T_i)\}$. This implies that $\{T'_1, \ldots, T'_m\}$ is also a schedule with minimum makespan. Note that the first t machine of schedule $\{T'_1, \ldots, T'_m\}$ contains more bags from $\{S_1, S_2, \ldots, S_{2t}\}$ than the first t machine of schedule $\{T_1, \ldots, T_m\}$. By repeating the above process, we can get a schedule with minimum makespan such that $S_1, S_2, \ldots, S_{2t}$ are all scheduled on the first t machines. This completes the proof of (1).

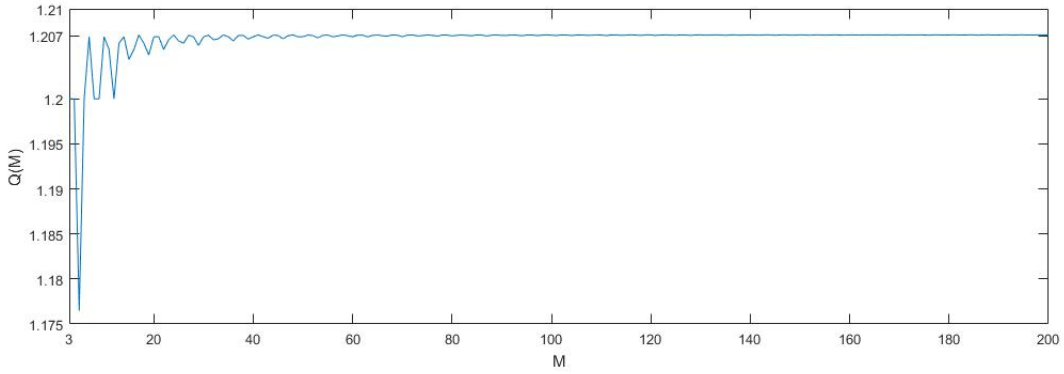

Figure 5.2: $Q(M)$ as a function of M

By choosing $m = M$, we know that $p(S_i) \leq k'\text{opt}_M = \dfrac{k's}{M}$ for any $1 \leq i \leq M$. For $t \leq M/2$, consider the schedule with minimum makespan on $m = M - t$ machines such that $S_1, S_2, \ldots, S_{2t}$ are scheduled on the first t machines. It follows that $\sum_{i=1}^{2t} p(S_i) \leq t \cdot k'\text{opt}_m = \dfrac{tk's}{M-t}$

Hence for any $t \leq M/2$, we have

$$\begin{aligned} s = \sum_{i=1}^{M} p(S_i) &= \sum_{i=1}^{2t} p(S_i) + \sum_{i=2t+1}^{M} p(S_i) \\ &\leq \frac{tk's}{M-t} + (M-2t) \cdot \frac{k's}{M} \end{aligned}$$

It follow that $k' \geq Q(M) = \max\limits_{t \leq \frac{M}{2}, t \in N} \dfrac{1}{\frac{t}{M-t} + \frac{M-2t}{M}}$. Let $L(t) = \dfrac{t}{M-t} + \dfrac{M-2t}{M}$. Then by taking the derivative, it is not hard to obtain that $\min\limits_{0 \leq t \leq \frac{M}{2}} L(t) = L((1 - \frac{\sqrt{2}}{2})M) = 2(\sqrt{2} - 1)$. And $Q(M) \approx \dfrac{1}{\min\limits_{0 \leq t \leq \frac{M}{2}} L(t)} = \dfrac{\sqrt{2}+1}{2} \approx 1.207$.

□

Figure 5.2 shows how $Q(M)$ changes with M.

5.2.1 Minimizing the Maximum Difference

Another objective we consider is to minimize the difference between the load of the most loaded and least loaded machines. This objective is particularly relevant to settings in which we want to achieve *fairness*. First of all, minimizing the maximum difference is a well-studied scheduling objective in

situations where fairness is important. Second, it models a type of fair allocation problem. Suppose that we want to split some goods among m people, where we don't know what m is in advance. In order to simplify the process, we can first divide the goods into M groups or bags, and then have the restriction that each person must take a subset of the bags.

For this problem, we consider a case where after packing the jobs into M bags, we are given $m \in [\alpha M, M]$ machines on which to schedule. Our bounds (see below) will depend on α, with, not surprisingly, better bounds for larger α. Since we assume every job is infinitesimal, the minimum difference between the load of the most loaded and least loaded machines is always 0 if we know the number of machines in advance. So we can not use the α-robust settings to evaluate our algorithms in this section and we introduce some new definitions here. Let s denote the sum of processing time of all jobs and let $A = s/M$. For a set of bags $S = \{S_1, S_2, \ldots, S_M\}$, we use $D_m(S)$ to denote the minimum difference between the load of the most loaded and least loaded machines when we schedule S on m machines. Our goal is to choose S so as to minimize $D(M, \alpha, S, A) = \max_{m \in [\alpha M, M]}\{D_m(S)\}$.

Assume that αM is an integer with value at most $M - 1$ and $M \geq 3$. Let $D^*(M, \alpha, A) = \min_S\{D(M, \alpha, S, A)\}$. We now give a lower bound of $D^*(M, \alpha, A)$.

5.2.3. *If $\alpha M > \frac{M}{2}$, then*

$$D^*(M, \alpha, A) \geq \frac{2(1-\alpha)M}{1 + (4\alpha + 1)(1-\alpha)M} \cdot A \ .$$

If $\alpha M \leq \frac{M}{2}$, then

$$D^*(M, \alpha, A) \geq \frac{2M^2 - 2M}{3M^2 + M - 2} \cdot A, \textit{ if } M \textit{ is odd;}$$

$$D^*(M, \alpha, A) \geq \frac{2M^2 - 4M}{3M^2 - 8} \cdot A, \textit{ if } M \textit{ is even.}$$

Proof. We may assume that $p(S_1) \leq p(S_2) \leq \cdots \leq p(S_M)$. For $i = 1, 2, \ldots, M$, let $A_i = \sum_{j=1}^{i} p(S_j)/i$. We first prove the following statement.

For $m > M/2$,

$$D_m(M, \alpha, S, A) \geq 2A_{2M-2m} - p(S_{2M-2m+1}). \tag{5.4}$$

It is sufficient to proof the following: there exists a scheduling to minimize the differences by schedule $S_1, S_2, \ldots, S_{2M-2m}$ on the first $M-m$ machines and schedule $S_{2M-2m+1}, \ldots, S_M$ to the rest of the $2m - M$ machines. Since then the maximum load on a machine is at least $\sum_{i=1}^{2M-2m} S_i/(M -$

$m) = 2A_{2M-2m}$ and the minimum load on a machine is at most $S_{2M-2m+1}$. Suppose not. Let the optimal way to schedule $\{S_i\}$ on m machines is by scheduling a set of bags, Y_i, on the i-th machine. We may assume that every Y_i contains at least one bag. Note that are at least $2m - M$ machine receive exactly one bag. We rename the machines so that $Y_{M-m+1}, Y_{M-m+2}, \ldots, Y_{M-m+(2m-M)} = Y_m$ contain only one bag. Suppose there exists $i \leq 2M - 2m$, $j \geq M - m + 1$ such that $Y_j = \{S_i\}$. Then there exists $i' \geq 2M - 2m + 1$, $j' \leq M - m$ such that $S_{i'} \in Y_{j'}$. Define $Y'_j = \{S_{i'}\}$, $Y'_{j'} = Y_{j'} \setminus S_{i'} \cup \{S_i\}$, and $Y'_\ell = Y_\ell$ for $\ell \neq j, j'$. Since $p(S_{i'}) \geq p(S_i)$, $p(Y'_{j'}) \geq p(Y_{j'})$. Note also that $p(Y'_j) \leq \max_{i=1}^{M}\{p(S_i)\} \leq \max_{i=1}^{M}\{p(Y_i)\}$. Hence $\max_{i=1}^{m}\{p(Y'_i)\} \leq \max_{i=1}^{m}\{p(Y_i)\}$. Since $p(Y'_{j'}) \geq p(S_i) = p(Y_j)$ and $p(Y'_j) = p(S_{i'}) \geq p(Y_j)$, $\min_{i=1}^{m}\{p(Y'_i)\} \geq \min_{i=1}^{m}\{p(Y_i)\}$. This implies that $\{Y'_i\}$ is also an optimal way of scheduling the bags. So we may assume that $\{Y_i\}$ satisfies that $\bigcup_{i=M-m+1}^{m} Y_i = \{S_{2M-2m+1}, S_{2M-2m+2} \ldots S_{2M-2m+2m-M} = S_M\}$. This completes the proof of (4).

Let $D = D^*(M, \alpha, A)$ and let $S = \{S_i\}$ be the set of bags that reaches the optima. First we assume that $\alpha M > \frac{M}{2}$. Let $A' = A_{2M-2\alpha M}$. By (5.4), $p(S_{(2-2\alpha)M+1}) \geq 2A' - D_{\alpha M}(S) \geq 2A' - D$. Since $D \geq D_M(S) = p(S_M) - p(S_1) \geq p(S_{(2-2\alpha)M+1}) - p(S_1)$, $p(S_1) \geq 2A' - 2D$ and $p(S_M) \geq p(S_{(2-2\alpha)M+1}) \geq 2A' - D$.

We know that $A_2 \geq p(S_1) \geq 2A' - 2D$. For $1 \leq k < (1-\alpha)M$, assume we know that $A_{2k} \geq 2A' - 2D + \frac{k-1}{2}(2A' - 3D)$. Then by (5.4), we have $p(S_{2k+1}) \geq 2A_{2k} - D_{M-k}(S) \geq 2(2A' - 2D + \frac{k-1}{2}(2A'-3D)) - D = 2A'-2D+k(2A'-3D)$. Also $p(S_{2k+2}) \geq p(S_{2k+1}) \geq 2A'-2D+k(2A'-3D)$. Therefore $A_{2k+2} \geq A_{2k}\frac{2k}{2k+2} + (2A' - 2D + k(2A' - 3D))\frac{2}{2k+2} \geq 2A' - 2D + \frac{k}{2}(2A' - 3D)$. Inductively, this implies that $A' = A_{2M-2\alpha M} \geq 2A' - 2D + \frac{(1-\alpha)M-1}{2}(2A' - 3D)$, which is equivalent to

$$D \geq \frac{2(1-\alpha)M}{1+3(1-\alpha)M}A'$$

Recall that $p(S_i) \leq p(S_1) + D \leq A' + D$ for any i, we have

$$\begin{aligned} MA = s &\leq 2(1-\alpha)MA' + (2\alpha - 1)M(A' + D) \\ &= MA' + (2\alpha - 1)MD \\ &\leq \frac{1+3(1-\alpha)M}{2(1-\alpha)}D + (2\alpha - 1)MD. \end{aligned}$$

This is equivalent to

$$D \geq \frac{2(1-\alpha)M}{1+(4\alpha+1)(1-\alpha)M} \cdot A \ .$$

For the case of $\alpha M \leq \frac{M}{2}$, if M is odd, then

$$D^*(M,\alpha,A) \geq D^*(M,\frac{M+1}{2M},A) \geq \frac{2M^2-2M}{3M^2+M-2} \cdot A \ .$$

If M is even, then

$$D^*(M,\alpha,A) \geq D^*(M,\frac{M+2}{2M},A) \geq \frac{2M^2-4M}{3M^2-8} \cdot A \ .$$

□

These bounds can most simply be parsed by saying that, if M is large then the lower bound is given by $\min\{2/3, 2/(4\alpha+1)\}A$. Note that when $\alpha \leq \frac{1}{2}$, the lower bound goes to $\frac{2}{3}A$ as M goes large. If M is even, then the bound of $\frac{2}{3}A$ can be reached by setting the bags as follows. Choose $S = \{S_1, S_2, \ldots, S_M\}$ such that $p(S_1) = p(S_2) = \cdots = p(S_{M/2}) = \frac{2}{3}A$ and $p(S_{M/2+1}) = p(S_{M/2+2}) = \cdots = p(S_M) = \frac{4}{3}A$, then it is easy to verify that $D_m(S) \leq \frac{2}{3}A$ for any $m \in [1, M]$.

For $\alpha > \frac{1}{2}$, we can set $D = \frac{(1-\alpha)M}{\alpha+(1-\alpha)(\alpha+1)M} \cdot A$ and $A' = \frac{1+3(1-\alpha)M}{2(1-\alpha)M} \cdot D$. Now we can choose the bags, $T = \{T_1, T_2, \ldots, T_M\}$, as follows. For $k \leq (1-\alpha)M$, set $p(T_{2k+1}) = p(T_{2k+2}) = 2A' - 2D + k(2A' - 3D)$; for $i \geq 2(1-\alpha)M + 1$, set $p(T_i) = 2A' - D$. The idea of this construction follows from the proof of Theorem 5.2.3, and by following this proof, it is not hard to verify that $D_m(S) \leq D$ for any $m \in [\alpha M, M]$. Note that $D' \approx \frac{1}{\alpha+1} \cdot A$ if M is large.

5.3 Scheduling discrete jobs

In this section we will provide an algorithm which, for any set of jobs, gives a $\frac{5}{3}+\epsilon$ robust ratio for $m \in [1, M]$. For ease of presentation, in this section we assume M is divisible by 4. In Appendix B, we sketch the changes that are needed when M is not divisible by 4.

Our algorithm will start by computing minimum makespan schedules for M, $3M/4$ and $M/2$ machines. We can use a PTAS for minimizing makespan on parallel machines to compute a $(1+\epsilon)$-approximate schedule. We use opt'_i to denote the value of the makespan obtained by running the PTAS on a scheduling instance with i machines. We will especially use $\text{opt}'_{M/2}$ and will denote this

value by b. We use $PTAS(i) = \{S_1, S_2, ..S_i\}$ to denote the result of the PTAS on a scheduling instance with i machines, where S_j is the set of jobs assigned to machine j.

Based on the values opt'_M, $\text{opt}'_{3M/4}$ and $\text{opt}'_{M/2}$, we will consider several cases. Each case will have the same structure. First, we will compute sets of jobs S_i from a PTAS. Second we will use *partitioning* routines to split the job set into M bags. We then learn the number of machines and need to schedule the bags on machines. In different cases, we will use different combinations of partitioning and scheduling routines. We then show that, for each case, the resulting schedule is $\frac{5}{3}(1+\epsilon)$ robust.

In Section 5.3.1, we will give our partitioning routines and prove some properties about each one. In Section 5.3.2, we will describe how to assemble the jobs into bags and then schedule them on machines.

5.3.1 Partitioning

In this subsection we will describe three useful partitioning algorithms, which will be used as subroutines in the main algorithm. Each one will take one or four sets of jobs and partition them into multiple bags with special properties, achieving some balance between the sizes of the bags and controlling the placement of large jobs. In all routines we will assume that we process the jobs in sorted size order, largest-to-smallest.

We begin with the first partitioning algorithm, Partition I, which partitions a job set into two bags such that neither of them is too big. This partitioning algorithm will be useful when we want to partition a job set "evenly". It implements the LPT algorithm, sorting the jobs in non-increasing order and then repeatedly placing the next job in the least loaded bag. It appears in Algorithm 3.

We now bound the sizes of the bags. Recall that the standard analysis of LPT shows that it is a $4/3 - 1/3m = 7/6$-approximation algorithm on 2 machines [25]. The bounds that we give are tight and are not implied by the standard analysis of LPT.

5.3.1. *Let (A, B) be the output of Partition I, then $p(A) = \max\{p(A), p(B)\} \leq \max\{p(j_1), 2b/3\}$. If the maximum is achieved by $p(j_1)$ then $A = \{j_1\}$.*

Proof. If j_1 has larger processing time than all the other jobs combined, i.e. $p(j_1) \geq \sum_{i=2}^{K} p(j_i)$, then for $i = 2, .., K$, LPT will put j_i in B and $\max\{p(A), p(B)\} = p(A) = p(j_1)$.

Algorithm 3 Partition $I(S)$

Input:

A set of jobs $S = \{j_1, j_2, .., j_K\}$ with $p(j_1) \geq p(j_2) \geq .. \geq p(j_K)$ and $p(S) \leq b$.

Main Process:

Run LPT on 2 machines, A and B, breaking ties in favor of A. If $p(A) \leq p(B)$, swap the names of A and B.

Output: (A, B).

Next consider the case when $p(j_1) < \sum_{i=2}^{K} p(j_i)$. We use d_t to denote the difference in the loads of A and B after adding job j_t (Note that $d_K = p(A) - p(B)$)ãĂĆ Pick k as the smallest integer such that $p(j_1) < \sum_{i=2}^{k} p(j_i)$, clearly $k \geq 3$. Then $p(j_k) \geq d_k = \sum_{i=2}^{k} p(j_i) - p(j_1)$ since $\sum_{i=2}^{k-1} p(j_i) \leq p(j_1)$. When we decide where to put j_{i+1}, the difference between the loads of A and B is d_i, and we put j_{i+1} in the bag with less load. Hence we must have $d_{i+1} \leq \max\{p(j_{i+1}), d_i\}$. Then inductively $d_K \leq \max\{p(j_K), p(j_{K-1}), .., p(j_{k+1}), p(j_k), d_k\}$. Because the jobs are indexed largest to smallest, and because $k \geq 3$, we can simplify the previous inequality to $d_K \leq p(j_3) \leq b/3$ (recall that $d_k \leq j_k$). Since $p(A) + p(B) = b$, it immediately follows that $\max\{p(A), p(B)\} \leq 2b/3$. □

Since we almost always need to put multiple bags on one machine, we want to create enough small bags to control the makespan. This demand leads to the next two partitioning algorithms, in which we partition job sets to bags such that half of them are small enough and the others are not too big. The first one, Partition II, works for the case when we have a job set with at most one large job. We will place the job with the largest processing time in set A and then fill set B greedily up to $b/3$ and then place the remaining jobs in A. The details appear in Algorithm 4.

5.3.2. *Let (A, B) be the output of Partition II, then $p(A) \leq 5b/6$ and $p(B) \leq b/3$.*

Proof. By line 2 clearly we have $p(B) \leq b/3$. Suppose, for a contradiction, that $p(A) > 5b/6$. Then $t \neq K$ and it follows that $p(j_1) + \sum_{i=t+1}^{K} p(j_i) > 5b/6$. Since $\sum_{i=1}^{K} p(j_i) \leq b$, we must have

Algorithm 4 Partition II(S)

Input:

A set of jobs $S = \{j_1, j_2, \ldots, j_K\}$ with $5b/6 \geq p(j_1) \geq p(j_2) \geq \cdots \geq p(j_K)$, $p(j_2) \leq b/3$ and $p(S) \leq b$.

Main Process:

1 $B = \emptyset$, $A = \{j_1\}$.

2 Let t be the largest integer such that $p(j_2) + p(j_3) + \cdots + p(j_t) \leq \frac{b}{3}$; put $j_2, \ldots, j_t$ in B.

3 Add $j_{t+1}, \ldots, j_K$ into A if $t \neq K$.

Output: (A, B).

$\sum_{i=2}^{t} p(j_i) = \sum_{i=1}^{K} p(j_i) - (p(j_1) + \sum_{i=t+1}^{K} p(j_i)) < b/6$.

But since $\sum_{i=2}^{t+1} p(j_i) > b/3$, we also have $p(j_{t+1}) = \sum_{i=2}^{t+1} p(j_i) - \sum_{i=2}^{t} p(j_i) > b/3 - b/6 = b/6$. Note that $t \geq 2$, hence $p(j_2) \geq p(j_{t+1}) > b/6 > \sum_{i=2}^{t} p(j_i)$, a contradiction. □

The last partitioning algorithm, Partition III, will handle the case when we have at least two large jobs and Partition II is not working. Specifically, we will take four job sets as input; three of them, S_1, S_3, S_4, have two large ($\geq b/3$) jobs and the other, S_2, has all small ($< b/3$) jobs. We partition them into eight bags such that we have four small bags and four bags that are not too big. The algorithm first puts the second biggest job from S_1 into A_1 and the remaining jobs from S_1 into A_2. It then greedily puts jobs from S_2 into A_1 and A_2 as long as they don't cause the load to go over $5b/6$. By greedily, here, we mean that it goes through the jobs in non-decreasing order of processing time, and if the job can fit on A_1 or A_2, we place it on one which it fits. We then use Partition I on the remaining jobs of S_2, running LPT to place jobs on B_1 and B_2. We also use Partition I to partition S_3 to (A_3, B_3), and S_4 to (A_4, B_4). We will show that either $p(B_1) + p(B_3)$ and $p(B_2) + p(B_4)$ are both not too big or we can switch jobs to ensure that neither is. The details of the algorithm appear in Algorithm 5.

Algorithm 5 Partition III(S_1, S_2, S_3, S_4)

Input:

Four sets of jobs: $S_1 = \{x_1, x_2, .., x_p\}$ such that $p(x_1) \geq p(x_2) \geq b/3$, $p(S_1) \leq b$, $S_2 = \{y_1, .., y_q\}$ such that $b/3 \geq p(y_1) \geq .. \geq p(y_q)$, $p(S_2) \leq b$, S_3 with $p(S_3) \leq b$ and S_4 with $p(S_4) \leq b$. Moreover for $i = 1, 2, 3, 4$, $p(j) \leq 2b/3$ for every $j \in S_i$.

Main Process:

1 $A_1 = \{x_2\}$. $A_2 = S_1 - \{x_2\}$, $i = 1$.

2 **while** ($(p(A_1) + p(y_i) \leq 5b/6)$ or $(p(A_2) + p(y_i) \leq 5b/6)$) and $(i \geq 1)$

3 if $p(A_1) + p(y_i) \leq 5b/6$, add y_i to A_1 else add y_i to A_2,

4 $i++$

5 Let S' be the jobs remaining in S_2.

6 Set (B_1, B_2) = Partition I(S'), (A_3, B_3) = Partition I(S_3), (A_4, B_4) = Partition I(S_4).

7 If $B_1 \cup B_2$ contains 3 jobs and B_2 contains only one job, let j_3 be the job with the least processing time in B_1. If $p(j_3) + p(A_3) \leq 5b/6$, move j_3 from B_1 to A_3; Else if $p(j_3) + p(A_4) \leq 5b/6$, move j_3 from B_1 to A_4; Else set $temp = B_1$, $B_1 = B_4$, $B_4 = temp$.

8 Else if $B_1 \cup B_2$ contains more than 3 jobs and B_2 contains one job. Let j' denotes the job in B_2 with the least processing time. Move j' from B_2 to A_3 if $p(j') + p(A_3) \leq 5b/6$.

Output: $(A_1, A_2, A_3, A_4, B_1, B_2, B_3, B_4)$.

5.3.3. *Let $(A_1, A_2, A_3, A_4, B_1, B_2, B_3, B_4)$ be the output of Partition III. These bags satisfy:*

1. $p(A_i) \leq 5b/6$, *for* $i = 1, 2, 3, 4$;
2. $p(B_i) \leq b/2$, *for* $i = 1, 2, 3, 4$;
3. $p(B_1) + p(B_3) \leq 5b/6$, $p(B_2) + p(B_4) \leq 5b/6$.

Proof. Let k_{ij} denote the load of jobs from S_i that are placed in bag A_j. Note that the jobs in bags B_1 and B_2 all come from S_2. Thus $k_{11} + k_{12} \leq b$ and $k_{21} + k_{22} + p(B_1) + p(B_2) \leq b$. For $i = 1, 2$, define the non-negative slack in each A_i as $\delta_i = 5b/6 - k_{1i} - k_{2i}$ and $\delta = \max\{\delta_1, \delta_2\}$.

Note that at line 1, $p(A_1) = p(x_2) \leq b/2$ and $p(A_2) = p(S_1) - p(x_2) \leq b - b/3 \leq 2b/3$. Since we call Partition I to separate S_3 and S_4, by Lemma 5.3.1, $p(A_3) \leq 2b/3$, $p(A_4) \leq 2b/3$ at line 6. Since for $i = 1, 2, 3, 4$, we will only add job to A_i as long as the load does not exceed $5b/6$, condition 1 holds.

We now consider conditions 2 and 3. By line 2, any job in $B_1 \cup B_2$ has load greater than δ. First consider the case when $B_1 \cup B_2$ contains at most two job. Then $p(B_1) \leq b/3$ and $p(B_2) \leq b/3$. Note that also $p(B_3) \leq p(S_3)/2 \leq b/2$ and similarly $p(B_4) \leq b/2$, hence the claim follows.

Next we consider the case when $B_1 \cup B_2$ contains $l \geq 3$ job before line 7. Note that,

$$\begin{aligned} p(B_1) + p(B_2) &\leq b - k_{21} - k_{22} \\ &= b - \left(\frac{5b}{6} - \delta_1 - k_{11}\right) - \left(\frac{5b}{6} - \delta_2 - k_{12}\right) \\ &\leq \frac{b}{3} + 2\delta \ . \end{aligned}$$

If both B_1 and B_2 contain at least 2 jobs, then $p(B_1) \geq 2\delta$ and $p(B_2) \leq b/3 + 2\delta - p(B_1) \leq b/3$. Similarly $p(B_2) \leq b/3$, hence the claim follows.

Next we consider the case when $l = 3$. If B_1 contains only one job, then $b/3 \geq p(B_1) \geq p(B_2)$ and the claim follows. We may assume $B_1 = \{j_2, j_3\}$, $B_2 = \{j_1\}$ and $p(j_1) \geq p(j_2) \geq p(j_3)$. Since $k_{11} = p(x_2) \leq b/2$ and $5b/6 - k_{11} \geq b/3 \geq p(y_1)$, we will always put y_1 in A_1. It implies that S_2 contains at least 4 jobs and $j_3 \leq p(S_2)/4 \leq b/4$. If in line 7 we move j_3 to either A_3 or A_4, then clearly claim follows since after line 7, both B_1 and B_2 contains one job and thus have load at most $b/3$. So we may assume that before line 7, $p(A_3) > 5b/6 - p(j_3) \geq 5b/6 - b/4 = 7b/12$, and then $p(B_3) \leq b - p(A_3) < 5b/12$. Similarly, we have $p(B_4) < 5b/12$. Hence $p(B_3) + p(B_4) < 5b/6$ before

line 7. Note also that $p(j_2) + p(j_3) \leq p(j_1) + p(y_1)$, hence $p(j_2) + p(j_3) \leq p(S_2)/2 \leq b/2$. So before line 7, $p(B_1) \leq b/2$ and $p(B_2) \leq b/3$. Recall that we will swap the name of B_1 and B_4, hence the claim follows.

The last case is when $l \geq 4$ and one of B_1 and B_2 contains only one job. If B_1 contains only one job, then $b/3 \geq p(B_1) \geq p(B_2)$ and the claim follows. We may assume that $B_2 = \{j_1\}$ and $B_1 = \{j_2, j_3, \ldots, j_l\}$ with $p(j_1) \geq p(j_2) \geq \cdots \geq p(j_l)$. Then by Partition I, $p(j_2) + \cdots + p(j_{l-1}) \leq p(B_1) \leq b/3$. Recall that $l \geq 4$, hence $p(j_l) \leq (p(j_2) + \cdots + p(j_{l-1}))/(l-2) \leq b/6$. By Lemma 5.3.1, $p(A_3) \leq 2b/3$. Thus $p(A_3) + p(j_k) \leq 5b/6$. So in line 8 we will always move j_k to A_3. After line 8, $p(B_2) = p(j_2) + \cdots + p(j_{l-1}) \leq b/3$. Hence the claim also follows. □

5.3.2 Packing and Scheduling

In the previous subsection, we gave different algorithms to partition jobs into bags. We will now show how to use these algorithms in conjunction with additional algorithms to schedule the bags (and thus jobs) onto machines. Recall that $\text{opt}'_{M/2}$ is the value of the makespan obtained by running the PTAS on a scheduling instance with $M/2$ machines and $b = \text{opt}'_{M/2}$. Two simple facts we will use throughout this section are that opt_m is non-increasing with m, the number of machine, and that $2\text{opt}_M \geq \text{opt}_{M/2}$, which implies that $\text{opt}_M \geq \frac{\text{opt}'_{M/2}}{2(1+\epsilon)} = \frac{b}{2(1+\epsilon)}$.

We now prove our bound of $\frac{5}{3} + \epsilon$ by considering three different cases. The first case is when $opt'_M \geq 3b/5$, which implies that the PTAS of the jobs on $M/2$ is good enough for M machines. We remark that if this case does not hold then there is no job with processing time greater than $3b/5$. The next case is when $opt'_{3M/4} \leq 4b/5$, which implies that we can start with $opt'_{3M/4}$ and do not need to split all job sets. The last and the hardest case is when $opt'_{3M/4} > 4b/5$ and $opt'_M < 3b/5$. In this case we will start with the PTAS on $M/2$ and split each job sets according to its structure. Before going to the detail of those three cases, we start with a lemma which gives a sufficient condition for bags to give a $\frac{5}{3}(1+\epsilon)$ robust ratio when the number of machine is less than $M/2$. It will be used in both the second and the last case.

5.3.4. *Let $\{S_1, \ldots, S_M\}$ denotes M bags such that for $i = 1, 2, \ldots, M/2$, $p(S_i) \leq 5b/6$, and for $i = M/2+1, M/2+2, \ldots, M$, $p(S_i) \leq 2b/3$. Then the following algorithm yields a schedule for the bags with a makespan of at most $5(1+\epsilon)\text{opt}_m/3$ for any $m \in [1, M/2)$.*

1 **if** $M/4 \leq m < M/2$. For $i = 1, 2, \ldots, m$, put S_i on machine i; for $i = m+1, m+2, \ldots, M/2$, put S_i on machine $i - m$; for $i = M/2+1, \ldots, M$, put S_i on the machine with the least load.

2 **if** $\frac{M}{2(k+1)} \leq m < \frac{M}{2k}$ for some integer $k \geq 2$, sort $\{S_1, S_2, \ldots, S_M\}$ in decreasing order according to their processing times. For $i = 1, \ldots, m$, put S_i on machine i; for $i = m+1, \ldots, 2m$, put S_i on machine $i - m$; for $i = 2m+1, \ldots, M$, put S_i on the machine with the least load.

Proof. First note that since $m < M/2$, $\text{opt}_m \geq \text{opt}_{M/2} \geq b/(1+\epsilon)$. Let machine r be the machine with the largest load and among the bags scheduled on machine r, S_t is the one with the largest index. For the case $M/4 \leq m < M/2$, if $t > M/2$, then we know $p(S_t) \leq 2b/3 \leq 2(1+\epsilon)\text{opt}_m/3$ and the makespan of our algorithm is at most $p(S_t) + \text{opt}_m \leq 5(1+\epsilon)\text{opt}_m/3$. Else $t \leq M/2$, then we know that we only put two bags on machine r, and therefore the makespan is at most $2 \cdot 5b/6 = 5b/3 \leq 5(1+\epsilon)\text{opt}_m/3$.

For the case $\frac{M}{2(k+1)} \leq m < \frac{M}{2k}$ with $k \geq 2$, if $t \leq 2m$, then similarly we know that we only put two bags on machine r, and therefore the makespan is at most $2 \cdot 5b/6 = 5b/3 \leq 5(1+\epsilon)\text{opt}_m/3$. So we may assume that $t \geq 2m+1$. Since $\sum_{i=1}^{2m} p(S_i) \geq \sum_{i=1}^{2m} p(S_t) = 2mp(S_t)$, $\text{opt}_m \geq \sum_{i=1}^{2m} p(S_i)/m \geq 2p(S_t)$. This implies that $p(S_t) \leq \text{opt}_m/2$ and the makespan of our algorithm is at most $p(S_t) + \text{opt}_m \leq 3\text{opt}_m/2$. □

5.3.2.1 Case I: $\text{opt}'_M \geq 3b/5$.

The first case is quite simple. We start with $PTAS(M/2)$, and partition each job set into two bags using Partition I. Those are our bags. For scheduling, if $m \geq M/2$, we just revert to the $M/2$ machine schedule, leaving the remaining machines empty. If $m < M/2$, we can run LPT to the bags on the machines. The details appear in Algorithm 6.

5.3.5. *If* $\text{opt}'_M \geq 3b/5$, *Algorithm 6 is* $\frac{5}{3}(1+\epsilon)$*-robust.*

Proof. For $m \geq M/2$,

$$\frac{ALG(m,M)}{\text{opt}_m} \leq \frac{\text{opt}'_{M/2}}{\text{opt}_M} \leq \frac{b}{\frac{\text{opt}'_M}{1+\epsilon}} \leq \frac{b}{\frac{3b/5}{1+\epsilon}} = (1+\epsilon)\frac{5}{3}.$$

For $m < M/2$, $\text{opt}_m \geq \text{opt}_{M/2} \geq b/(1+\epsilon)$. We may assume the bags are $S'_1, \ldots, S'_M$ such that $p(S'_1) \geq p(S'_2) \cdots \geq p(S'_M)$. Let k be the number such that for $i \leq k$, $p(S'_i) > 2b/3$ and for $i > k$, $p(S'_i) \leq 2b/3$. Recall that each bag S'_i comes from running Partition I on some job set S_j with

Algorithm 6 Algorithm for Case I

Packing:

1 Let $S = PTAS(M/2) = \{S_1, \ldots, S_{M/2}\}$.

2 **for** $k = 1 \ldots, M/2$, let $(A_k, B_k) =$ Partition I(S_k).

3 Return the M bags $\{A_1, \ldots, A_{M/2}, B_1, \ldots, B_{M/2}\}$.

Scheduling:

1 **if** $m \geq M/2$, schedule A_i and B_i on machine i for $i \leq M/2$. Leave the remaining machines empty.

2 **if** $m < M/2$, run LPT to schedule the bags on m machines.

$p(S_j) \leq b$. Hence by Lemma 5.3.1, either $p(S'_i) \leq 2b/3$ or it is a bag with a single job. Specifically, $S'_1, \ldots, S'_k$ all contain only one job. Let the jobs be $j_1, j_2, \ldots, j_k$ respectively. Let machine r be the machine with the largest load and among the bags scheduled on machine r, S_t is the one with the largest index. If $t > k$, then we know $p(S_t) \leq 2b/3 \leq 2(1+\epsilon)\text{opt}_m/3$. By the property of LPT, the makespan of our algorithm is at most $p(S_t) + \text{opt}_m \leq 5(1+\epsilon)\text{opt}_m/3$. If $t \leq k$, then the makespan of our algorithm is equal to the makespan of running LPT on jobs $\{j_1, j_2, \ldots, j_k\}$, which is at most 4/3 times the minimum makespan of scheduling $\{j_1, j_2, \ldots, j_k\}$ on m machines [25]. Since the minimum makespan of scheduling $\{j_1, j_2, \ldots, j_k\}$ on m machines is at most opt_m, our makespan in this case is at most $4\text{opt}_m/3$.

□

5.3.2.2 Case II: $\text{opt}'_M < \frac{3b}{5}$ and $\text{opt}'_{3M/4} \leq 4b/5$.

In this case, we will use the PTAS schedule for $3M/4$ machines as input to Partition I. For $S_i \in PTAS(3M/4)$, we call S_i *bad* if S_i contains a job with processing time at least $\frac{2}{3}\text{opt}'_{3M/4}$. For the packing step, We will run Partition I on $M/4$ job sets, try to avoid a bad set if possible and then schedule the bags on machines. For the scheduling step, we will have several cases, based on how many bad machines that we have. The details appear in Algorithm 7.

5.3.6. *If* $\text{opt}'_M < \frac{3b}{5}$ *and* $\text{opt}'_{3M/4} \leq 4b/5$ *then Algorithm 7 is* $\frac{5}{3}(1+\epsilon)$*-robust.*

Algorithm 7 Algorithm for Case II

Packing:

1 Let $S = PTAS(3M/4) = \{S_1, .., S_{3M/4}\}$. Let v be the number of bad sets in S. Rename the sets so that $S_1, \ldots, S_v$ are bad sets.

2 For $M/2 < k \leq 3M/4$, let $(A_k, B_k) =$ Partition I(S_k).

3 Return the M bags $\{S_1, S_2, .., S_{M/2}, A_{M/2+1}, \ldots, A_{3M/4}, B_{M/2+1}, .., B_{3M/4}\}$.

Scheduling:

1 **if** $m \geq 3M/4$, schedule S_i on machine i for $i \leq M/2$, and schedule A_j, B_j on machine j for $j > M/2$. Leave the remaining machines empty.

2 **if** $v > M/2$ and $v \leq m < 3M/4$, first put $S_1, S_2, \ldots, S_{M/2}, A_{M/2+1} \cup B_{M/2+1}, \ldots, A_v \cup B_v$ on v machines respectively, then schedule the rest of the bags with at most one in each machine.

3 **if** $v > M/2$ and $M/2 \leq m < v - 1$, or $v \leq M/2$ and $M/2 \leq m < 3M/4$, schedule the bags arbitrarily such that each machine contains at most two bags and at most one bag comes from $\{S_1, \ldots, S_{M/2}\}$.

4 **if** $m < M/2$, we follow the strategy in Lemma 5.3.4.

Proof. For $m \leq 3M/4$, the load on each machine is at most $\text{opt}'_{3M/4} \leq 4b/5$, hence we have,

$$\frac{ALG(m,M)}{\text{opt}_m} \leq \frac{\text{opt}'_{3M/4}}{\text{opt}_M} \leq \frac{4b/5}{\frac{b}{2(1+\epsilon)}} < (1+\epsilon)\frac{5}{3}.$$

By Lemma 5.3.1, for $k > \max\{v, M/2\}$, $p(B_k) \leq p(A_k) \leq \frac{2}{3}\text{opt}'_{3M/4}$. If $v > M/2$, and $v \leq m < 3M/4$ or $v \leq M/2$ and $M/2 \leq m < 3M/4$, the load on each machine is at most $\text{opt}'_{3M/4} + \max_{i>v}\{p(A_i)\} \leq \frac{5}{3}\text{opt}'_{3M/4}$, so we have,

$$\frac{ALG(m,M)}{\text{opt}_m} \leq \frac{\frac{5}{3}\text{opt}'_{3M/4}}{\text{opt}_{3M/4}} \leq (1+\epsilon)\frac{5}{3}.$$

If $v > M/2$ and $M/2 \leq m < v-1$, then since the number of jobs with processing time at least $\frac{2}{3}\text{opt}'_{3M/4}$ is at least v, $\text{opt}_{v-1} \geq \frac{4}{3}\text{opt}'_{3M/4}$. On the other hand, each machine has load at most $2\text{opt}'_{3M/4}$. So we obtain

$$\frac{ALG(m,M)}{\text{opt}_m} \leq \frac{2\text{opt}'_{3M/4}}{\text{opt}_{v-1}} \leq \frac{2\text{opt}'_{3M/4}}{\frac{4}{3}\text{opt}'_{3M/4}} < (1+\epsilon)\frac{5}{3}\ .$$

Since $\text{opt}'_M < 3b/5$, every job has processing time at most $3b/5$. Notice that for $i \leq M/2$ and $j > M/2$, $p(S_i) \leq \text{opt}'_{3M/4} \leq 4b/5$ and $p(B_j) \leq p(A_j) \leq \max\{3b/5, 2\text{opt}'_{3M/4}/3\} = 3b/5 < 2b/3$, we can apply Lemma 5.3.4 on the case $m < M/2$. □

5.3.2.3 Case III: $\text{opt}'_M < \frac{3b}{5}$ and $\text{opt}'_{3M/4} > 4b/5$.

This final case is the most complicated one. We say a job is *big* if it has processing time at least $b/3$. Given a set of jobs, we call the set *2-big* if it has two big jobs. We call the set *1-big* if it is not 2-big and has one big job. We call a set that is neither 1-big nor 2-big, *0-big*. Give a set S of job sets, we let $v_i(S)$ denote the number of i-big sets and we assume that we have functions *i-big(S)* which return an i-big set from S, assuming one exists. For a job set S_i, we use $S_i(j)$ to denote the jth largest job in set S_i.

The main idea for packing here is to use the optimal schedule for $M/2$ machines, but to split the jobs assigned to each machine into 2 bags. For each S_i, we want to partition it into two sets such that one is small enough and the other is not too big. We can use Partition II to achieve this goal, if and only if S_i contains at most one big job, that is, S_i is 0-big or 1-big. If S_i is 2-big and thus contain two big jobs, we try to group three 2-big job sets with one 0-big job set, and then

partition them into eight sets such that four are small enough and four not too big, using Partition III. Roughly speaking, if many S_i contain two big jobs, then we can give a good lower bound on opt, else we will have a good partition. The details appear in Algorithm 8.

5.3.7. *If* $\text{opt}'_M < 3b/5$ *and* $\text{opt}'_{3M/4} > 4b/5$*, Algorithm 8 is* $\frac{5}{3}(1+\epsilon)$*-robust.*

Proof. Since $\text{opt}'_M < 3b/5$, every job has processing time at most $3b/5$. For $k = 1, 2, 3, \ldots, 4u$, by Lemma 5.3.2, we have $p(A_k) \leq 5b/6$ and $p(B_k) \leq b/2$ and if moreover $k = 4t+1$ or $4t+2$ for an integer t, $p(B_k) + p(B_{k+2}) \leq 5b/6$ and we call such B_k and B_{k+2} *paired bags*. For $k = 4u+1, 4u+2, .., 4u+w$, we partition a 2-big job set S_i into $\{A_k, B_k\}$ by putting the second biggest job in B_k and the rest in A_k. Hence $b/3 \leq p(B_k) \leq b/2$, $p(A_k) = p(S_i) - p(B_k) \leq b - b/3 = 2b/3$. For $k = 4u+w+1, 4u+w+2, .., \frac{M}{2}$, by Lemma 5.3.3 we have $p(A_k) \leq 5b/6, p(B_k) \leq b/3$ (note that after line 3 there is no 2-big job set left and hence we can use Partition II in line 4).

Next consider the scheduling process in Algorithm 4. First consider the case $m \geq \max\{3M/4, M/2+w+2u\}$. For $i \leq M/2$, the load of each machine is at most $\max_i\{p(A_i)\} \leq 5b/6$. For $i = M/2+1, M/2+2, \ldots, M/2+2u$, we always schedule a paired bags on the machines, so the load is still at most $5b/6$. For the rest of the machines, we schedule $B_{4u+1}, \ldots, B_{M/2}$ on them such that at most two bags on each machines and if there are two bags, at most one of them is from $B_{4u+1}, \ldots, B_{4u+w}$, hence the load is at most $b/3 + b/2 = 5b/6$. Therefore

$$\frac{ALG(m,M)}{\text{opt}_m} \leq \frac{5b/6}{\text{opt}_M} \leq \frac{5b/6}{\frac{b}{2(1+\epsilon)}} = (1+\epsilon)\frac{5}{3}.$$

Next consider the case $w+2u > M/4$ and $3M/4 \leq m < M/2+w+2u$. Since $4u \leq v_2(S)+v_0(S) \leq M/2$, $2u \leq M/4$ and $v_2(S) - 3u = w \geq 1$. If $u = \lfloor v_2(S)/3 \rfloor < v_0(S)$, since $v_2(S) \geq 3u+1$, $v_2(S) = 3u+1$ or $3u+2$. If $v_2(S) = 3u+1$, then $M/4 < w+2u = v_2(S) - u = 2u+1$ and $M/4 \leq 2u$, but $M/2 \geq v_2(S) + v_0(S) > 3u+1+u$, a contradiction. If $v_2(S) = 3u+2$, then $M/4 < v_2(S) - u = 2u+2$ and $M/4 \leq 2u+1$, but $M/2 \geq v_2(S) + v_0(S) > 3u+2+u$, a contradiction. So we have $u \geq v_0(S)$ and $w+2u = v_2(S) - u \leq v_2(S) - v_0(S)$. Note that in total we have at least $2v_2(S) + v_1(S) = M/2 + v_2(S) - v_0(S)$ jobs with processing time at least $b/3$. Hence $\text{opt}_{M/2+w+2u-1} \geq 2b/3$. On the other hand in this case the load on each machine is at most $\max\{\max_i\{A_i\}, 2\max_i\{B_i\}\} \leq b$.

$$\frac{ALG(m,M)}{\text{opt}_m} \leq \frac{b}{\text{opt}_{M/2+w+2u-1}} \leq \frac{b}{2b/3} < (1+\epsilon)\frac{5}{3}\ .$$

Algorithm 8 Algorithm for Case III

Packing:

1 Let $S = PTAS(\frac{M}{2}) = \{S_1, .., S_{\frac{M}{2}}\}$, $u = \min\{\left\lfloor \frac{v_2(S)}{3} \right\rfloor, v_0(S)\}$, $w = \max\{v_2(S) - 3u, 0\}$ and $S' = S$.

2 **if** $u \geq 1$, **for** $k = 1, 5, 9, \ldots, 4u - 3$,

$C = 2\text{-big}(S'); D = 0\text{-big}(S'); E = 2\text{-big}(S'); F = 2\text{-big}(S')$;

$(A_k, \ldots, A_{k+3}, B_k, \ldots, B_{k+3}) = \text{Partition III}(C, D, E, F)$;

$S' = S' - C - D - E - F$.

3 **if** $w \geq 1$, **for** $k = 4u + 1, 4u + 2, \ldots, 4u + w$,

$C = 2\text{-big}(S'); B_k = \{C(2)\}; A_k = C/B_k; S' = S' - C$.

4 **for** $k = 4u + w + 1, 4u + w + 2, \ldots, \frac{M}{2}$.

Let C be any set in S'. $(A_k, B_k) = \text{Partition II(C)}$; $S' = S' - C$.

5 Return the bags $\{A_1, A_2, \ldots, A_{\frac{M}{2}}, B_1, \ldots, B_{\frac{M}{2}}\}$.

Scheduling:

1 **if** $m \geq \max\{3M/4, M/2 + w + 2u\}$

for $i \leq M/2$, schedule A_i on machine i;

for $i = M/2 + 1, M/2 + 3, \ldots, M/2 + 2u - 1$, schedule B_{2i-M-1} and B_{2i-M+1} on machine i and B_{2i-M} and B_{2i-M+2} on machine $i + 1$;

for $i = M/2 + 2u + 1, \ldots, m$, schedule the rest of the bags on those machine such that each machine contains at most two bags and if there are two bags on a machine, at most one of them is from $B_{4u+1}, \ldots, B_{4u+w}$;

2 **if** $w + 2u > M/4$ and $3M/4 \leq m < M/2 + w + 2u$

for $i \leq M/2$, schedule A_i on machine i;

for $i = M/2 + 1, \ldots, m$, schedule the rest of the bags on the machines so that each machine contains at most two bags;

3 **if** $M/2 \leq m < 3M/4$

Schedule all bags amongst the machines so that there are at most two bags on each machine and if there are 2 bags on a machine, at most one of them is A_i for some i.

4 **if** $m < M/2$, we follow the strategy in Lemma 5.3.4.

In the case $M/2 \leq m < 3M/4$, the load on each machine is at most $\max_i\{A_i\} + \max_i\{B_i\} \leq 5b/6 + b/2 = 4b/3$. Therefore we have

$$\frac{ALG(m,M)}{\mathrm{opt}_m} \leq \frac{4b/3}{\mathrm{opt}_{3M/4}} \leq (1+\epsilon)\frac{4b/3}{4b/5} = (1+\epsilon)\frac{5}{3}\ .$$

For the case when $m < M/2$, by Lemma 5.3.4 the claim follows. □

By combining Lemma 5.3.5, 5.3.6 and 5.3.7, we can deduce the following lemma. The running time comes from calling the PTAS [34] 3 times, various sorting and heap data structure operations needed to implement LPT and the other algorithms. The M term is in the running time because of the need to perform operations on the M bags.

5.3.8. *When M is divisible by 4, Algorithm 6, 7, 8 together gives a $\frac{5}{3}(1+\epsilon)$-robust algorithm with running time of $O((M+n)\log n)$, where the hidden constant depends exponentially on $1/\epsilon$.*

Recall that the algorithm uses the terms $M/2$ and $3M/4$ and assumes that they are integers. When M is not divisible by 4, the algorithm and analysis still work with some small changes.

5.3.9. *When M is not divisible by 4, with some small changes, Algorithm 6, 7, 8 still work with a $\frac{5}{3}(1+\epsilon)$ robust ratio.*

Proof. We first consider the case when $M = 4r+2$, for some nonnegative integer r. Then we can still follow Algorithm 6. For Algorithm 8, we replace every $M/4$ by $r+1$ and every $3M/4$ by $3r+2$. And then Lemma 5.3.7 still holds with the same argument under the assumption that $opt'_{3r+2} > 4b/5$ instead of $opt'_{3M/4} > 4b/5$. So the only remaining case is $opt'_{3r+2} \leq 4b/5$ and we can follow Algorithm 7 with small modifications. In the packing phase of Algorithm 7, we will only partition r sets and return with $M = 4r+2$ bags $\{S_1, \ldots, S_{2r+2}, A_{2r+3}, \ldots, A_{3r+2}, B_{2r+3}, \ldots, B_{3r+2}\}$. Then in the scheduling phase, line 1 holds for $m \geq 3r+2$, line 2 holds for $v > 2r+2$ and $v \leq m < 3r+2$, and line 3 holds for $v > 2r+2$ and $2r+2 \leq m < v-1$, or $v \leq 2r+2$ and $2r+2 \leq m < 3r+2$. The corresponding part of Lemma 5.3.6 holds accordingly. The last case is when $m \leq 2r+1$, and we can follow Lemma 5.3.4 with the same strategy except for replacing $M/2$ with $2r+2$ and $M/4$ with $r+1$.

Next we consider the case when $M = 4r+1$, for some nonnegative integer r. In this case we denote $b = opt'_{2r+1}$ instead of $opt'_{M/2}$ and we replace $M/2$ by $2r+1$ in all three algorithm. Then we

still have that $opt_M \geq \frac{b}{1+\epsilon}$. For Algorithm 6, in the packing phase, we start with $S = PTAS(2r+1)$ in line 1 and we run partition I for $S_1, S_2 \ldots, S_{2r}$ in line 2. Then we can still return M bags $\{A_1, \ldots, A_{2r}, B_1, \ldots, B_{2r}, S_{2r+1}\}$ in line 3. We may assume that either S_{2r+1} contains a job with processing time greater than $2b/3$ or every job has processing time at most $2b/3$. For the case $m \geq 2r+1$, we follow line 1 of the scheduling phase of Algorithm 6 and clearly the robust ratio still holds. Next we consider the case for $m \leq 2r$, and we run LPT to schedule the bags on m machines. Recall that for $i = 1, 2, \ldots, 2r$, either $p(A_i) \leq 2b/3$ or A_i is a bag with a single job, and same for B_i. To prove the ratio still hold, it is sufficient to show that the bags finish last has size at most $2b/3$. If S_{2r+1} does not contain a job with processing time greater than $2b/3$, then every job has processing time at most $2b/3$. We may assume that we run LPT with S_{2r+1} first (otherwise $p(S_{2r+1}) \leq 2b/3$) and that S_{2r+1} is not the bag finishes last since otherwise the makespan is at most b. But now the bags finishes last has size at most $2b/3$ and the claim holds. So we may assume S_{2r+1} contains a job with processing time greater than $2b/3$, say j. Then the proof of Lemma 5.3.5 works except for the case when the machine finishes last contains only bags with single jobs and S_{2r+1}. But its makespan is at most $4opt_m/3 + p(S_{2r+1} \setminus \{j\}) \leq 5opt_m/3$. For Algorithm 8, recall that we do it under the case when $opt_{4r+1} < 3b/5 < 2b/3$, then it implies that for the sets in $\{S_1, S_2, \ldots, S_{2r+1}\} = PTAS(2r+1)$, at most $2r$ of which contain at least 2 big jobs. As we can only provide $4r+1$ bag, in the packing phase, we will return with the bags $\{A_1, \ldots, A_{2r+1}, A' = B_1 \cup B_3, B_2, B_4, B_5, \ldots, B_{2r+1}\}$ if $u \geq 1$ and $\{A_1, \ldots, A_{2r+1}, A' = B_1 \cup B_{2r+1}, B_2, B_3, \ldots, B_{2r}\}$ otherwise. Then $p(A') \leq 5b/6$ and we will treat it like an A-bag. Then in the scheduling phase, line 1 holds for $m \geq \max\{3r+2, 2r+1+w+2u\}$, line 2 holds for $w+2u > r+1$ and $3r+2 \leq m < 2r+1+w+2u$, and line 3 holds for $2r+2 \leq m \leq 3r+1$. The corresponding part of Lemma 5.3.7 holds accordingly under the assumption that $opt_{3r+1} > 4b/5$. Also for the case when $m \leq 2r+1$, we can follow Lemma 5.3.4 with the same strategy except for replacing $M/2$ with $2r+2$ and $M/4$ with $r+1$. It left the case when $opt_{3r+1} \leq 4b/5$ and we follow Algorithm 7 with modifications. In the packing phase of Algorithm 7, we will only partition r sets and return with $M = 4r+1$ bags $\{S_1, \ldots, S_{2r+1}, A_{2r+2}, \ldots, A_{3r+1}, B_{2r+2}, \ldots, B_{3r+1}\}$. Then in the scheduling phase, line 1 holds for $m \geq 3r+1$, line 2 holds for $v > 2r+1$ and $v \leq m < 3r+1$, and line 3 holds for $v > 2r+1$ and $2r+1 \leq m < v-1$, or $v \leq 2r+1$ and $2r+1 \leq m < 3r+1$. The corresponding part of Lemma 5.3.6 holds accordingly. Also for the case when $m \leq 2r+1$, we can follow Lemma 5.3.4 similarly.

For the last case when $M = 4r + 3$, for some nonnegative integer r. In this case we denote $b = opt'_{2r+2}$ instead of $opt_{M/2}$ and we replace $M/2$ by $2r+2$ in all three algorithm. The modification of Algorithm 6 is exactly the same as in the case when $M = 4r + 1$. For Algorithm 8, similarly we have at most $2r + 1$ sets in $PTAS(2r + 2)$ which contain at least 2 big jobs. in the packing phase, we will return with the bags $\{A_1, \ldots, A_{2r+2}, A' = B_1 \cup B_3, B_2, B_4, B_5, \ldots, B_{2r+2}\}$ if $u \geq 1$ and $\{A_1, \ldots, A_{2r+2}, A' = B_1 \cup B_{2r+2}, B_2, B_3, \ldots, B_{2r+1}\}$ otherwise. Then in the scheduling phase, line 1 holds for $m \geq \max\{3r + 3, 2r + 2 + w + 2u\}$, line 2 holds for $w + 2u > r + 1$ and $3r + 3 \leq m < 2r + 2 + w + 2u$, and line 3 holds for $2r + 3 \leq m \leq 3r + 2$. The corresponding part of Lemma 5.3.7 holds accordingly. Also for the case when $m \leq 2r + 2$, we can follow Lemma 5.3.4 with the same strategy except for replacing $M/2$ with $2r + 3$ and $M/4$ with $r + 2$. It left the case when $opt_{3r+2} \leq 4b/5$ and we follow Algorithm 7 with similar modifications as in the previous case. This finishes the proof of 5.3.9. □

Bibliography

[1] S. Albers. Recent advances for a classical scheduling problem. In *Proceedings of the Automata, Languages and Programming*, pages 4–14, 2013.

[2] S. Albers and M. Hellwig. Online makespan minimization with parallel schedules. In *Proceedings of the Scandinavian Symposium and Workshops on Algorithm Theory*, pages 13–25, 2014.

[3] S. Albers and G. Schmidt. Scheduling with unexpected machine breakdowns. *Discrete Applied Mathematics*, 110:85–89, 2001.

[4] J. A. Aslam, A. Rasala, C. Stein, and N. E. Young. Improved bicriteria existence theorems for scheduling. In *Proceedings of the Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 846–847, 1999.

[5] B. Aspvall, M.F. Plass, and R.E. Tarjan. A linear-time algorithm for testing the truth of certain quantified boolean formulas. *Inf. Process. Lett.*, 8(3):121–123, 1979.

[6] F. Bonomo, M. Chudnovsky, P. Maceli, O. Schaudt, M. Stein, and M. Zhong. Three-coloring and list three-coloring graphs without induced paths on seven vertices. to appear in Combinatorica, `http://doi.org/10.1007/s00493-017-3553-8`, 2017.

[7] D. Bruce, C.T. Hoàng, and J. Sawada. A certifying algorithm for 3-colorability of P_5-free graphs. In *Proceedings of the 20th International Symposium on Algorithms and Computation*, pages 594–604. Springer-Verlag, 2009.

[8] E. Camby and O. Schaudt. A new characterization of P_k-free graphs. *Algorithmica*, 75(1):205–217, 2016.

[9] L. Chen, N. Megow, R. Rischke, and L. Stougie. Stochastic and robust scheduling in the cloud. In *Proceedings of the APPROX-RANDOM*, pages 175–186, 2015.

[10] M. Chudnovsky, J. Goedgebeur, O. Schaudt, and M. Zhong. Obstructions for three-coloring graphs with one forbidden induced subgraph. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA*, pages 1774–1783, 2016.

[11] M. Chudnovsky, J. Goedgebeur, O. Schaudt, and M. Zhong. Obstructions for three-coloring and list three-coloring h-free graphs. Submitted for publication, 2017.

[12] M. Chudnovsky, S. Spirkl, and M. Zhong. Four-coloring p_6-free graphs. i. extending an excellent precoloring. Submitted for publication, 2018.

[13] M. Chudnovsky, S. Spirkl, and M. Zhong. Four-coloring p_6-free graphs. ii. finding an excellent precoloring. Submitted for publication, 2018.

[14] D. Corneil, Y. Perl, and L. Stewart. Cographs: recognition, applications and algorithms. *Congressus Numerantium*, 43:249–258, 1984.

[15] Y. Disser, M. Klimm, N. Megow, and S. Stiller. Packing a knapsack of unknown capacity. In *Proceedings of the Symposium on Theoretical Aspects of Computer Science*, pages 276–287, 2014.

[16] K. Edwards. The complexity of colouring problems on dense graphs. *Theoretical Computer Science*, 43:337–343, 1986.

[17] L. Epstein, A. Levin, A. Marchetti-Spaccamela, N. Megow, J. Mestre, M. Skutella, and L. Stougie. Universal sequencing on an unreliable machine. *SIAM Journal on Computing*, 41:565–586, 2012.

[18] P. Erdős. Graph theory and probability. *Canadian Journal of Mathematics*, 11:34–38, 1959.

[19] P. Erdős, A. Rubin, and H. Taylor. Choosability in graphs. *Congressus Numerantium*, 26:125–157, 1979.

[20] F. Galvin, I. Rival, and Sands. B. A ramsey-type theorem for traceable graphs. *Journal of Combinatorial Theory, Series B*, 33(1):7–16, 1982.

[21] J. Goedgebeur. Homepage of generator for obstructions against list-3 colorability: `http://caagt.ugent.be/listcriticalpfree/`.

[22] P. Golovach, M. Johnson, D. Paulusma, and J. Song. A Survey on the Computational Complexity of Colouring Graphs with Forbidden Subgraphs. arXiv:1407.1482v6 [cs.CC], June 2015.

[23] P. Golovach, D. Paulusma, and J. Song. Closing complexity gaps for coloring problems on H-free graphs. *Information and Computation*, 237:204–214, 2014.

[24] P.A. Golovach, M. Johnson, D. Paulusma, and J. Song. A survey on the computational complexity of coloring graphs with forbidden subgraphs. *Journal of Graph Theory*, 84(4):331–363, 2017.

[25] R. L. Graham. Bounds for certain multiprocessing anomalies. *Bell System Technical Journal*, pages 1563–1581, 1966.

[26] R. Hassin and S. Rubinstein. Robust matchings. *SIAM Journal on Discrete Mathematics*, 15:530–537, 2002.

[27] P. Hell and S. Huang. Complexity of coloring graphs without paths and cycles. *Discrete Applied Mathematics*, 216, Part 1:211–232, 2017.

[28] C.T. Hoàng, M. Kamiński, V.V. Lozin, J Sawada, and X. Shu. Deciding k-colorability of P_5-free graphs in polynomial time. *Algorithmica*, 57:74–81, 2010.

[29] C.T. Hoàng, B. Moore, D. Recoskie, J. Sawada, and M. Vatshelle. Constructions of k-critical P_5-free graphs. *Discrete Applied Mathematics*, 182:91–98, 2015.

[30] I. Holyer. The NP-completeness of edge-coloring. *SIAM Journal on Computing*, 10:718–720, 1981.

[31] S. Huang. Improved complexity results on k-coloring P_t-free graphs. In *Proceedings of the International Symposium on Mathematical Foundations of Computer Science 2013*, volume 7551 of *Lecture Notes in Computer Science*, pages 551–558, 2013.

[32] S. Huang, M. Johnson, and D. Paulusma. Narrowing the complexity gap for colouring (C_s, P_t)-free graphs. arXiv:1407.1480v1 [cs.CC], July 2014.

[33] B.M.P. Jansen and S. Kratsch. Data reduction for graph coloring problems. *Information and Computation*, 231:70–88, 2013.

[34] K. Jansen, K. M. Klein, and J. Verschae. Closing the gap for makespan scheduling via sparsification technique. In *Proceedings of the Automata, Languages and Programming*, pages 1–13, 2016.

[35] M. Kamiński and V.V. Lozin. Coloring edges and vertices of graphs without short or long cycles. *Contributions to Discrete Mathematics*, 2:61–66, 2007.

[36] R. Karp. Reducibility among combinatorial problems. In R. Miller and J. Thatcher, editors, *Complexity of Computer Computations*, pages 85–103. Plenum Press, New York, 1972.

[37] D. Král', J. Kratochvíl, Zs. Tuza, and G.J. Woeginger. Complexity of coloring graphs without forbidden induced subgraphs. In A. Brandstadt and V.B. Le, editors, *Proceedings of the International Workshop on Graph-Theoretic Concepts in Computer Science 2001*, volume 2204 of *Lecture Notes in Computer Science*, pages 254–262, 2001.

[38] F. Lazebnik and V.A. Ustimenko. Explicit construction of graphs with an arbitrary large girth and of large size. *Discrete Applied Mathematics*, 60:275–284, 1995.

[39] D. Leven and Z. Galil. NP-completeness of finding the chromatic index of regular graphs. *Journal of Algorithms*, 4:35–44, 1983.

[40] F. Maffray and G. Morel. On 3-colorable P_5-free graphs. *SIAM Journal on Discrete Mathematics*, 26:1682–1708, 2012.

[41] F. Maffray and M. Preissmann. On the NP-completeness of the k-colorability problem for triangle-free graphs. *Discrete Mathematics*, 162:313–317, 1996.

[42] J. Matuschke, M. Skutella, and J. A. Soto. Robust randomized matchings. In *Proceedings of the Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1904–1915, 2015.

[43] N. Megow. Robustness and approximation for universal sequencing. *Gems of Combinatorial Optimization and Graph Algorithms*, pages 133–141, 2015.

[44] S. Mellin. Polynomielle farbungsalgorithmen für P_k-freie graphen. Diplomarbeit am Institut für Informatik, Universität zu Köln, 2002.

[45] J. Niño-Mora. Stochastic scheduling. In C.A. Floudas and P.M. Pardalos, editors, *Encyclopedia of Optimization*, pages 3818–3824. Springer, New York, 2009.

[46] L. K. Platzman and J. J. Bartholdi. Spacefilling curves and the planar travelling salesman-problem. *Journal of the ACM*, 36:719–737, 1989.

[47] A. Pokrovskiy. private communication.

[48] F.P. Ramsey. On a problem of formal logic. *Proceedings of the London Mathematical Society*, 30:264–286, 1930.

[49] B. Randerath and I. Schiermeyer. 3-Colorability $\in$ P for P_6-free graphs. *Discrete Applied Mathematics*, 136:299–313, 2004.

[50] B. Randerath, I. Schiermeyer, and M. Tewes. Three-colorability and forbidden subgraphs. II: polynomial algorithms. *Discrete Mathematics*, 251:137–153, 2002.

[51] A. Rasala, C. Stein, E. Torng, and P. Uthaisombut. Existence theorems, lower bounds and algorithms for scheduling to meet two objectives. In *Proceedings of the Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 723–731, 2002.

[52] D. Scheische. On a property of the class of n-colorable graphs. *Journal of Combinatorial Theory, Series B*, 16(2):191–193, 1974.

[53] P. Seymour. Barbados workshop on graph coloring and structure, 2014.

[54] D. B. Shmoys and M. Sozio. Approximation algorithms for 2-stage stochastic scheduling problems. In *Proceedings of the Integer Programming and Combinatorial Optimization International Conference*, pages 145–157, 2007.

[55] J. Stacho, 2014. Personal communication.

[56] C. Stein and M. Zhong. Scheduling when you don't know the number of machines. In *Proceedings of the Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1261–1273, 2018.

[57] L. Stockmeyer. Planar 3-colorability is polynomial complete. *ACM SIGACT News*, 5(3):19–25, 1973.

[58] Zs. Tuza. Graph colorings with local constraints – a survey. *Discussiones Mathematicae. Graph Theory*, 17:161–228, 1997.

[59] V. Vizing. Coloring the vertices of a graph in prescribed colors. *Metody Diskretnogo Analiza*, 29:3–10, 1976.

[60] E. Zhang. Bounded obstructions for three-coloring graphs with lists of size two, 2017. Senior Thesis, Princeton University.