


Complexity of Stability

Fabian Frei 

Department of Computer Science, ETH Zürich, Switzerland
fabian.frei@inf.ethz.ch

Edith Hemaspaandra 

Department of Computer Science, Rochester Institute of Technology, NY, USA
eh@cs.rit.edu

Jörg Rothe 

Institut für Informatik, Heinrich-Heine-Universität Düsseldorf, Germany
rothe@hhu.de

Abstract

Graph parameters such as the clique number, the chromatic number, and the independence number are central in many areas, ranging from computer networks to linguistics to computational neuroscience to social networks. In particular, the chromatic number of a graph (i.e., the smallest number of colors needed to color all vertices such that no two adjacent vertices are of the same color) can be applied in solving practical tasks as diverse as pattern matching, scheduling jobs to machines, allocating registers in compiler optimization, and even solving Sudoku puzzles. Typically, however, the underlying graphs are subject to (often minor) changes. To make these applications of graph parameters robust, it is important to know which graphs are stable for them in the sense that adding or deleting single edges or vertices does not change them. We initiate the study of stability of graphs for such parameters in terms of their computational complexity. We show that, for various central graph parameters, the problem of determining whether or not a given graph is stable is complete for Θ_2^P , a well-known complexity class in the second level of the polynomial hierarchy, which is also known as “parallel access to NP.”

2012 ACM Subject Classification Theory of computation → Problems, reductions and completeness; Theory of computation → Complexity classes; Mathematics of computing → Extremal graph theory

Keywords and phrases Stability, Robustness, Complexity, Local Modifications, Colorability, Vertex Cover, Clique, Independent Set, Satisfiability, Unfrozenness, Criticality, DP, coDP, Parallel Access to NP

Digital Object Identifier 10.4230/LIPIcs.ISAAC.2020.19

Related Version A full version of this paper, which includes the appendices, is published on arXiv [16] and available at <https://arxiv.org/abs/1910.00305>.

Funding *Edith Hemaspaandra*: Research done in part while on sabbatical at Heinrich-Heine-Universität Düsseldorf and supported in part by NSF grant DUE-1819546 and a Renewed Research Stay grant from the Alexander von Humboldt Foundation.

Jörg Rothe: Research supported by DFG grants RO 1202/14-2 and RO 1202/21-1.

Acknowledgements We thank the anonymous referees for their careful reading of this paper and suggestions for improvement.

1 Introduction

In this first section, we motivate our research topic, introduce the necessary notions and notation, and provide an overview of both the related work and our contribution.



© Fabian Frei, Edith Hemaspaandra, and Jörg Rothe;
licensed under Creative Commons License CC-BY

31st International Symposium on Algorithms and Computation (ISAAC 2020).

Editors: Yixin Cao, Siu-Wing Cheng, and Minming Li; Article No. 19; pp. 19:1–19:14

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

1.1 Motivation

The following extends an example given by Bollobás [3] (see also [26, Section 2.3.3]). Consider a graph whose vertices represent the ISAAC-2020 attendees, and an edge between any two vertices representing the wish of these two researchers to attend each other’s talks. The ISAAC-2020 organizers have elicited this information in advance, wishing to ensure that every participant can attend all the desired talks and also give their own presentation. Therefore, they color this graph, where each color represents a time slot (running parallel sessions within each time slot). What is the smallest number of colors needed so that no two adjacent vertices have the same color, i.e., what is the chromatic number of this graph? Suppose it is 12; so 12 time slots are enough to make all the participants happy. Now, however, the ISAAC-2020 organizers receive messages from Professor Late and Professor Riser expressing their wish to attend each other’s talks as well. Does this additional edge increase the chromatic number of the graph, requiring an additional time slot? Or does it always remain the same; that is, is this graph stable with respect to the chromatic number and adding edges?

Informally stated, a graph is *stable with respect to some graph parameter* (such as the chromatic number) if some type of small perturbation of the graph (a local modification such as adding an edge or deleting a vertex) does not change the parameter. Other graph parameters we consider are the clique number, the independence number, and the vertex cover number. This notion of stability formalizes the robustness of graphs for these parameters, which is important in many applications. Typical applications of the chromatic number, for instance, include coloring algorithms for complex networks such as social, economic, biological, and information networks (see, e.g., Jackson’s book on social and economic networks [26] or Khor’s work on applying graph coloring to biological networks [28]). In particular, social networks can be colored to find roles [15] or to study human behavior in small controlled groups [27, 10]. In various applied areas of computer science, graph coloring has also been used for register allocation in compiler optimization [7], pattern matching and pattern mining [37], and scheduling tasks [29]. To ensure that these applications of graph parameters are robust, graphs need to be stable for them with respect to certain operations. We initiate a systematic study of stability of graphs in terms of their computational complexity.

1.2 Notions and Notation

In this subsection, we define the core notions used in this paper and fix our notation.

1.2.1 Complexity Classes

We begin with the relevant complexity classes. Besides P, NP, and coNP, these are DP, coDP, and Θ_2^P . The class DP, introduced by Papadimitriou and Yannakakis [33], is the second level of the Boolean hierarchy over NP; that is, $DP = NP \wedge coNP = \{L_1 \cap L_2 \mid L_1 \in NP \wedge L_2 \in coNP\}$ is the set of all intersections of NP languages with coNP languages. Equivalently, it can be seen as the *differences* of NP languages, whence the name. An example of a trivially DP-complete language is $SAT-UNSAT = SAT \times UNSAT$, where UNSAT is the set of all unsatisfiable CNF-formulas. The complement class coDP contains exactly the unions of NP languages with coNP languages.

The class Θ_2^P , whose name is due to Wagner [39], belongs to the second level of the polynomial hierarchy; it can be defined as $\Theta_2^P = P^{NP[\mathcal{O}(\log n)]}$, which is the class of problems that can be solved in polynomial time by an algorithm with access to an oracle that decides arbitrary instances for an NP-complete problem – with one instance per call and each such query taking constant time – restricted to a logarithmic number of queries. (Without the last

restriction, we would get the class $\Delta_2^P = P^{NP}$.) Results due to Hemachandra [21, Thm. 4.10] usefully characterize Θ_2^P as P_{tt}^P , the class of languages that are polynomial-time truth-table reducible to NP. By definition, this is the same as P_{\parallel}^{NP} , the class of languages that are polynomial-time recognizable with *unlimited parallel* access to an NP oracle. *Unlimited* means that an algorithm witnessing the membership of a problem in P_{\parallel}^{NP} can query the oracle on as many instances of an NP-complete problem as it wants – which due the polynomial running-time means at most polynomially many – while *parallel* means that all queries need to be sent simultaneously. The characterization of Θ_2^P as $P^{NP[\mathcal{O}(\log n)]}$, in contrast, allows the logarithmically many queries to be *adaptive*; that is, they can be sent interactively, with one depending on the oracle’s answers to the previous ones. Membership proofs for Θ_2^P are usually easy; we will see a simple example of how to give one at the beginning of Section 4.

Note that the definitions immediately yield the inclusions $NP \cup \text{coNP} \subseteq DP \subseteq \Theta_2^P \subseteq \Delta_2^P$.

1.2.2 Graphs and Graph Numbers

Throughout this paper graphs are simple. Let \mathcal{G} be the set of all (simple) graphs and \mathbb{N} the set of natural numbers including zero. For any set M , we denote its *cardinality* or *size* by $\|M\|$. A map $\xi: \mathcal{G} \rightarrow \mathbb{N}$ is called a *graph number*. In this paper, we examine the prominent graph numbers α , β , χ , and ω , which give the size of a maximum independent set, the size of a minimum vertex cover, the size of a minimum coloring (i.e., the minimum number of colors allowing for a proper vertex coloring), and the size a maximum clique, respectively.

Let V , E , and \bar{E} be the functions that map a graph G to its vertex set $V(G)$, its edge set $E(G)$, and its set of *nonedges* $\bar{E}(G) = \{\{u, v\} \mid u, v \in V(G) \wedge u \neq v\} - E(G)$, respectively.

Let G and H be graphs. We denote by $G \cup H$ the disjoint union and by $G + H$ the *join*, which is $G \cup H$ with all *join edges* – i.e., the edges $\{v, w\} \in V(G) \times V(H)$ – added to it.¹

For $v \in V(G)$, $e \in E(G)$, and $e' \in \bar{E}(G)$, we denote by $G - v$, $G - e$, and $G + e'$ the graphs that result from G by deleting v , deleting e , and adding e' , respectively.

For any $k \in \mathbb{N}$, we denote by I_k and K_k the empty (i.e., edgeless) and complete graph on k vertices, respectively. The graph $I_0 = K_0$ without any vertices is called the *null graph*. A vertex v is *universal* with respect to a graph G if it is adjacent to all vertices $V(G) - \{v\}$.

1.2.3 Stability

Let G be a graph. An edge $e \in E(G)$ is called *stable* with respect to a graph number ξ (or ξ -*stable*, for short) if $\xi(G) = \xi(G - e)$, that is, deleting e leaves ξ unchanged. Otherwise (that is, if the deletion of e does change ξ), e is called ξ -*critical*. For a vertex $v \in V(G)$ instead of an edge $e \in E(G)$, stability and criticality are defined in the same way.

A graph is called ξ -stable if all of its edges are ξ -stable. A graph whose vertices – rather than edges – are all ξ -stable is called ξ -vertex-stable. The notions of ξ -criticality and ξ -vertex-criticality are defined analogously. Note that each edge and vertex is either stable or critical, whereas a graph might be neither. An unspecified ξ defaults to the chromatic number χ .

¹ We adopt the notation $G + H$ for the join from Harary’s classical textbook on graph theory [18, p. 21].

A traditional term for stability with respect to adding edges and vertices – rather than deleting them – is *unfrozenness*.² Specifically, a nonedge $e \in \overline{E}(G)$ is called *unfrozen* if adding it to the graph G leaves χ unchanged, and *frozen* otherwise. All of these notions extend naturally to vertices (where we can freely choose to which existing vertices a new vertex is adjacent, implying an exponential number of possibilities), to entire graphs, and to any graph number ξ , as just seen for stability and criticality.

We call a graph *two-way stable* if it is both stable and unfrozen, everything with respect to the chromatic number and deleting an edge as the default choice. Again, we have the analogous set of notions with respect to vertices and any graph number ξ .

Prefixing a natural number $k \in \mathbb{N}$ to any of these notions additionally requires the respective graph number to be exactly k . For example, a graph G is *k-critical* if and only if $\chi(G) = k$ and $\chi(G - e) \neq k$ for every $e \in E(G)$.

The notion of stability can be naturally applied to Boolean formulas as well. We call a formula Φ in conjunctive normal form *stable* if deleting an arbitrary clause C does not change its satisfiability status – that is, if it either is satisfiable (and of course stays so upon deletion of a clause) or if it and all its 1-clause-deleted subformulas $\Phi - C$ are unsatisfiable.

1.2.4 Languages

We denote by CNF the set of formulas in conjunctive normal form and by 3CNF, 4CNF, and 6CNF the set of CNF-formulas with *exactly* 3, 4, and 6 *distinct* literals per clause, respectively.³ The sets SAT and 3SAT contain the satisfiable, UNSAT and 3UNSAT the unsatisfiable formulas from CNF and 3CNF, respectively. Let $\text{STABLEUNSAT} = \{\Phi \in \text{UNSAT} \mid (\Phi - C) \in \text{UNSAT} \text{ for every clause } C \text{ of } \Phi\}$ be the set of stably unsatisfiable formulas. The set $\text{STABLECNF} = \text{SAT} \cup \text{STABLEUNSAT}$ consists of the stable CNF-formulas. Intersecting with 3CNF yields the classes STABLE3UNSAT and STABLE3CNF and so on.

Let STABILITY be the set of stable graphs and UNFROZENNESS the set of unfrozen graphs, both with respect to the default graph number χ . The set of two-way stable graphs is $\text{TWOWAYSTABILITY} = \text{STABILITY} \cap \text{UNFROZENNESS}$. Once more, these definitions extend naturally. For example, 4-VERTEXSTABILITY is the set of (with respect to the default χ) 4-vertex-stable graphs and β -TWOWAYSTABILITY consists of the graphs for which the vertex-cover number β remains unchanged upon deletion or addition of an edge.

1.2.5 AND Functions and OR Functions

Following Chang and Kadin [8], we say that a language $L \subseteq \Sigma^*$ has AND_2 if there is a polynomial-time computable function $f: \Sigma^* \times \Sigma^* \rightarrow \Sigma^*$ such that for all $x_1, x_2 \in \Sigma^*$, we have $x_1 \in L \wedge x_2 \in L \iff f(x_1, x_2) \in L$. If this is the case, we call f an AND_2 function for L . If

² The notion of instance parts being either frozen or unfrozen has originally been introduced to the field of computational complexity in analogy to the physical process of freezing [30, 31]. The sudden shift from P to NP-hardness that can be observed when transitioning from 2SAT to 3SAT by allowing a larger and larger percentage of clauses of length 3 rather than 2, for example, mimics the phase transition from liquid to solid, with the former granting much higher degrees of freedom to the substance's constituents than the latter. Based on this general intuition, Beacham and Culberson [2] then more formally defined the notion of unfrozenness with regard to an arbitrary graph property that is downward monotone (meaning that a graph keeps the property when edges are deleted); they call a graph unfrozen if it also keeps the property when an arbitrary new edge is added. We naturally extend this notion to arbitrary graph numbers, which are not necessarily monotone.

³ In the literature, these set names are often prefixed by an E, emphasizing the exactness. This is notably not the case for a paper by Cai and Meyer [6] that contains a construction crucially relying on this restriction. We will build upon this construction later on and are thus bound to the same constraint.

there even is a polynomial-time computable function $f : \bigcup_{k=0}^{\infty} (\Sigma^*)^k \rightarrow \Sigma^*$ such that for every $k \in \mathbb{N}$ and for all $x_1, \dots, x_k \in \Sigma^*$ we have $x_1 \in L \wedge \dots \wedge x_k \in L \iff f(x_1, \dots, x_k) \in L$, then we say that L has AND_{ω} . Replacing \wedge with \vee , we get the analogous notions OR_2 and OR_{ω} . Note that a language has AND_2 if and only if its complement has OR_2 , with the analogous statement holding for AND_{ω} and OR_{ω} .

1.3 Related Work

Many interesting problems are suspected to be complete for either DP or Θ_2^P . While membership is usually trivial in these cases, matching lower bounds are rare and hard to prove. For example, Woeginger [41] observes that determining whether a graph has a wonderfully stable partition is in Θ_2^P , and leaves it as an open problem to settle the exact complexity. Wagner, who introduced the class name Θ_2^P [38], provided a number of hardness results for variants of standard problems such as Satisfiability, Clique and Colorability, which are designed to be complete for DP or Θ_2^P . For example, he proves the DP-completeness of $\text{EXACTCOLORABILITY} = \{(G, k) \in \mathcal{G} \times \mathbb{N} \mid \chi(G) = k\}$ [38, Thm. 6.3.1 with $k = 1$] and the Θ_2^P -completeness of $\text{ODDVERTEXCOVER} = \{G \in \mathcal{G} \mid \beta(G) \text{ is odd}\}$ [38, Thm. 6.1.2].⁴ He obtains the analogous results for Colorability, Clique [38, Thm. 6.3], Independent Set instead of Vertex Cover [38, Thm. 6.4] and points out [38, second-to-last paragraph] that his proof techniques also yield the Θ_2^P -completeness of the equality version of all of these problems – for example, $\text{EQUALVERTEXCOVER} = \{(G, H) \in \mathcal{G}^2 \mid \beta(G) = \beta(H)\}$. The same holds true for the comparison versions such as $\text{COMPAREVERTEXCOVER} = \{(G, H) \in \mathcal{G}^2 \mid \beta(G) \leq \beta(H)\}$.⁵ The DP-completeness of EXACTCOLORABILITY has been extended to the subproblem of recognizing graphs with chromatic number 4 [34]. Furthermore, a few election problems have been proved to be Θ_2^P -complete by Hemaspaandra et al. [22, 23], by Rothe et al. [35], and Hemaspaandra et al. [24].

In general, establishing lower bounds proved to be difficult for many natural DP-complete and particularly Θ_2^P -complete problems. Consequently, hardness results remained rather rare in the area of criticality and stability, despite the great attention that these natural notions have garnered from graph theorists ever since the seminal paper by Dirac [13] from 1952; see for example the classical textbooks by Harary [18, chapters 10 and 12] and Bollobás [3, chapter IV] – the latter having a precursor dedicated exclusively to extremal graph theory [4, chapters I and V] – and countless papers over the decades, of which we cite some selected examples from early to recent ones [14, 19, 1, 40, 17, 20, 12, 25, 11]. A pioneering complexity result by Papadimitriou and Wolfe [32, Thm. 1] establishes the DP-completeness of MINIMAL-UNSAT . (They call a formula minimally unsatisfiable if deleting an arbitrary clause renders it satisfiable, that is, if it is critical.) They also proved that determining, given a graph G and a $k \in \mathbb{N}$, whether G is k - ω -vertex-critical is a DP-complete problem [32, Thm. 4]. Later, Cai and Meyer [6] showed the DP-completeness of k - VERTEXCRITICALITY (which they call $\text{MINIMAL-}k\text{-UNCOLORABILITY}$) for all $k \geq 3$. Burjons et al. [5] recently extended this result to the more difficult case of edge deletion, showing that k - CRITICALITY is DP-complete for all $k \geq 3$ [5, Thm. 8]. They also provided the first Θ_2^P -hardness result for a criticality problem, namely for β - VERTEXCRITICALITY [5, Thm. 15]. Note the drop in difficulty down to DP when fixing the graph number. This emerges as a general pattern, as evidenced by our results outlined in the contribution section below.

⁴ Note that Wagner originally derived his results with respect to the more restricted form of polynomial-time reducibility via Boolean formulas, indicated by the *bf* in the class name. He later proved the resulting notions to be equivalent, however; that is, we have $\text{PP}_{\text{bf}}^P = \Theta_2^P$ [39].

⁵ Spakowski and Vogel explicitly proved the Θ_2^P -completeness of $\text{COMPAREVERTEXCOVER}$ [36, Thm. 12], COMPARECLIQUE and $\text{COMPAREINDEPENDENTSET}$ [36, Thm. 13]. For other cases, see Appendix Q [16].

Stability, in contrast to criticality, has been sorely neglected by the computational complexity community, which is surprising in light of its apparent practical relevance – for example in the design of infrastructure, where stability is a most desirable property. A very small exception to this are Beacham and Culberson [2], who proved a comparably easy variant of Unfrozenness, namely $\{(G, k) \mid \chi(G) \leq k \text{ and } G \text{ is unfrozen}\}$, to be NP-complete.

1.4 Contribution

We choose four of the most prominent graph problems – Colorability, Vertex Cover, Independent Set, and Clique – to analyze the complexity of stability. We prove all of them to be Θ_2^P -complete for the default case of edge deletion. For unfrozenness – that is, stability with respect to edge addition – we prove the same, with the one exception of Colorability. For this problem, we prove that the existence of a construction with a few simple properties would be sufficient to prove Θ_2^P -completeness. Finally, we introduce the notion of two-way stability – stability with respect to both deleting and adding edges – and prove again Θ_2^P -completeness for all four problems. Table 1 provides an overview of these results, showcasing surprising contrasts between some of the problems.

We also derive several other useful results with broad appeal on their own, among these being the coDP-completeness of STABLE3CNF [Thm. 14], the DP-completeness of k -STABILITY and k -VERTEXSTABILITY for all $k \geq 4$ [Thm. 18], general criteria for proving DP-hardness [Lems. 33 and 34], and finally constructions such as the edge-stabilizing gadget [Lem. 19] that yields an AND_ω function for STABILITY [Cor. 20] and has potential applications in various contexts such as reoptimization and general graph theory.

■ **Table 1** An overview of our results regarding the complexity of different stability problems. See Section 3 for the results on Clique and Independent Set; almost all of them follow in analogy to the ones for Vertex Cover, with α -VERTEXSTABILITY and ω -VERTEXSTABILITY being the exception.

With respect to this base problem and graph number:	Stability		Unfrozenness		Two-Way Stability	
	Edge	Vertex	Edge	Vertex	Edge	Vertex
<i>Vertex Cover, β</i>	[Thm. 23] Θ_2^P -compl.	[Thm. 21] P	[Thm. 25] Θ_2^P -compl.	[Thm. 24] P	[Thm. 31] Θ_2^P -compl.	[Thm. 28] P
<i>Independent Set, α and Clique, ω</i>	Θ_2^P -compl.	Θ_2^P -compl.	Θ_2^P -compl.	P	Θ_2^P -compl.	P
<i>Colorability, χ</i>	Θ_2^P -compl. [Thm. 7]	Θ_2^P -compl. [Thm. 8]	? [Thm. 26]	P [Thm. 24]	Θ_2^P -compl. [Thm. 29]	P [Thm. 28]

2 Basic Observations

We begin with a few very basic and useful observations that will be used implicitly and, where appropriate, explicitly throughout the paper. The proofs are given in Appendix A [16].

► **Observation 1.** *The deletion of an edge or of a vertex either decreases the chromatic number by exactly one or leaves it unchanged.*

► **Observation 2.** *Let $e = \{u, v\}$ be a critical edge. Then u and v are critical as well.*

► **Observation 3.** *Let v be a stable vertex. Then all edges incident to v are stable.*

► **Observation 4.** *Let G be a graph. A vertex $v \in V(G)$ is critical if and only if there is an optimal coloring of G that assigns v a color with which no other vertex is colored.*

3 Connections between Clique, Vertex Cover, and Independent Set

As is to be expected, the three problems of Clique, Vertex Cover, and Independent Set are so closely related that almost all stability results for one of them carry over to the other two in a straightforward way. We state the connections in Proposition 5, proved in Appendix B [16].

► **Proposition 5.** *Let \overline{G} denote the complement graph of G . We have the following equalities.*

1. β -STABILITY = α -STABILITY = $\{\overline{G} \mid G \in \omega$ -UNFROZENNESS $\}$.
2. β -UNFROZENNESS = α -UNFROZENNESS = $\{\overline{G} \mid G \in \omega$ -STABILITY $\}$.
3. β -TWOWAYSTABILITY = α -TWOWAYSTABILITY = $\{\overline{G} \mid G \in \omega$ -TWOWAYSTABILITY $\}$.
4. β -VERTEXSTABILITY = $\{I_n \mid n \in \mathbb{N}\}$.
5. α -VERTEXSTABILITY = $\{\overline{G} \mid G \in \omega$ -VERTEXSTABILITY $\}$.
6. β -VERTEXUNFROZENNESS = β -VERTEXTWOWAYSTABILITY = $\{K_0\}$.
7. α -VERTEXUNFROZENNESS = α -VERTEXTWOWAYSTABILITY = ω -VERTEXUNFROZENNESS = ω -VERTEXTWOWAYSTABILITY = \emptyset .

An interesting inversion in this pattern occurs for the vertex deletion case. Here, switching from β to α or ω in fact flips the stability problem to the criticality version and vice versa.

► **Proposition 6.** *We have the following equalities.*

1. β -VERTEXSTABILITY = α -VERTEXCRITICALITY = $\{\overline{G} \mid G \in \omega$ -VERTEXCRITICALITY $\}$.
2. β -VERTEXCRITICALITY = α -VERTEXSTABILITY = $\{\overline{G} \mid G \in \omega$ -VERTEXSTABILITY $\}$.

Proposition 6 is proved in Appendix C [16]. Using it, we directly obtain from the Θ_2^P -hardness of β -VERTEXCRITICALITY [5] the same for α -VERTEXSTABILITY and, by complementing the graphs, ω -VERTEXSTABILITY. Unfortunately, β -VERTEXCRITICALITY is the only problem to yield any nontrivial result via the connection between stability and criticality.

We now turn our attention to the remaining stability problems, for which the hardness proofs will require substantially more effort.

4 Stability and Vertex-Stability for Colorability

We will prove Θ_2^P -completeness for both STABILITY and VERTEXSTABILITY.

► **Theorem 7.** *Determining whether a graph is stable is Θ_2^P -complete.*

► **Theorem 8.** *Determining whether a graph is vertex-stable is Θ_2^P -complete.*

As is typical, the upper bounds are immediate: We can determine the chromatic number of a graph and all its 1-vertex-deleted and 1-edge-deleted subgraphs with a polynomial number of parallel queries to an oracle for the standard, NP-complete colorability problem $\{(G, k) \in \mathcal{G} \times \mathbb{N} \mid \chi(G) \leq k\}$. Specifically, the queries (G, k) , $(G - e, k)$, and $(G - v, k)$ for every $e \in E(G)$, every $v \in V(G)$, and every $k \in \{0, \dots, \|V(G)\|\}$ suffice to determine whether G is stable and whether it is vertex-stable. To prove the matching lower bounds, we first note that the lower bound for Theorem 8 implies the lower bound for Theorem 7.

► **Lemma 9.** *VERTEXSTABILITY polynomial-time many-one reduces to STABILITY.*

It can be shown that mapping a graph G to its self-join $G + G$ provides the required reduction. Due to the space restrictions, the proof of Lemma 9 is deferred to Appendix D [16].

It remains to establish the lower bound of Theorem 8, that is, to prove that determining whether a graph is vertex-stable is Θ_2^P -hard. Proving Θ_2^P -hardness is not easy. However, we will now argue that it suffices to show that VERTEXSTABILITY is coDP-hard.

Chang and Kadin [9, Thm. 7.2] show that a problem is Θ_2^P -hard if it is DP-hard and has OR_ω . Observing that Θ_2^P is closed under complement, we obtain the following corollary.

► **Corollary 10.** *If a coDP-hard problem has AND_ω , then it is Θ_2^P -hard.*

We note that the join is an AND_ω function for VERTEXSTABILITY; see Appendix E [16]. Now, Theorem 8 follows from Corollary 10 and the coDP-hardness of VERTEXSTABILITY.

► **Theorem 11.** *The join is an AND_ω function for VERTEXSTABILITY and UNFROZENNESS.*

► **Lemma 12.** *Determining whether a graph is vertex-stable is coDP-hard.*

To prove Lemma 12, we show in Theorem 14 that $\text{STABLE3CNF} = 3\text{SAT} \cup \text{STABLE3UNSAT}$ is coDP-complete and then reduce it to VERTEXSTABILITY in Theorem 17. We will use twice the following lemma, whose straightforward proof is deferred to Appendix F [16].

► **Lemma 13.** *There is a polynomial-time many-one reduction from SAT to 3SAT converting a CNF-formula Φ into a 3CNF-formula Ψ such that Φ is stable if and only if Ψ is stable.*

► **Theorem 14.** *STABLE3CNF is coDP-complete.*

Proof. It is immediate that STABLE3CNF is in coDP. To show coDP-hardness, we will show that STABLE3CNF is coNP-hard, NP-hard, and has OR_2 . The coDP-hardness then follows by applying an observation by Chang and Kadin [9, Lem. 5] – a set is DP-hard if it is NP-hard, coNP-hard, and has an AND_2 function – to the complement language.

coNP-hardness. It is easy to see that the function $f: \Phi \mapsto \Phi \wedge (x \vee y \vee z) \wedge (x \vee y \vee \bar{z}) \wedge (x \vee \bar{y} \vee z) \wedge (x \vee \bar{y} \vee \bar{z}) \wedge (\bar{x} \vee y \vee z) \wedge (\bar{x} \vee y \vee \bar{z}) \wedge (\bar{x} \vee \bar{y} \vee z) \wedge (\bar{x} \vee \bar{y} \vee \bar{z})$, where x, y , and z are fresh variables not occurring in Φ , reduces 3UNSAT to STABLE3CNF.

NP-hardness. We give a reduction from 3SAT to STABLE4CNF; composing it with the reduction from Lemma 13 yields the desired reduction to STABLE3CNF. Given a 3CNF-formula $\Phi = C_1 \wedge \dots \wedge C_m$ over $X = \{x_1, \dots, x_n\}$, map it to the 4CNF-formula $\Psi = (C_1 \vee y) \wedge (C_1' \vee y') \wedge (C_1'' \vee y'') \wedge \dots \wedge (C_m \vee y) \wedge (C_m' \vee y') \wedge (C_m'' \vee y'') \wedge (\bar{y} \vee \bar{y}' \vee \bar{y}'')$, where the clauses C_i' and C_i'' are just like the clauses C_i but with a new copy of variables $X' = \{x_1', \dots, x_n'\}$ and $X'' = \{x_1'', \dots, x_n''\}$ instead of X , respectively, and y, y' , and y'' being three fresh variables as well. Deleting the clause $(\bar{y} \vee \bar{y}' \vee \bar{y}'')$ renders Ψ trivially satisfiable; any assignment that sets y, y' and y'' to 1 will do. Thus Ψ is stable if and only if it is satisfiable. It remains to prove the equisatisfiability of Φ and Ψ .

First assume that Φ has a satisfying assignment $\sigma: X \rightarrow \{0, 1\}$. Then Ψ is satisfied by any assignment τ with $\tau(x_i') = \tau(x_i'') = \sigma(x_i)$ for $i \in \{1, \dots, n\}$ and $\tau(y) = 0$. Now assume that Ψ has a satisfying assignment τ . Then Φ is satisfied by $\sigma: x_i \mapsto \tau(x_i)$ if $\tau(y) = 0$, by $\sigma': x_i \mapsto \tau(x_i')$ if $\tau(y') = 0$, and by $\sigma'': x_i \mapsto \tau(x_i'')$ if $\tau(y'') = 0$.

OR₂. In their proof of DP-completeness, Papadimitriou and Wolfe [32, Lem. 3 plus corollary] implicitly gave a simple AND_2 function for both MINIMALUNSAT and MINIMAL3UNSAT (the sets of unsatisfiable formulas that become satisfiable after deleting any clause). We make use of the same construction and defer the full proof to Appendix G [16].

This concludes the proof that STABLE3CNF is coDP-complete. ◀

All that is left to do is to reduce STABLE3CNF to VERTEXSTABILITY. First, we consider the known reduction from MINIMAL3UNSAT to VERTEXMINIMAL3UNCOLORABILITY by Cai and Meyer [6]. It maps a formula Φ with m clauses C_1, \dots, C_m to a graph G_Φ , whose vertex set includes, among others, a vertex called v_s and, for every $i \in \{1, \dots, m\}$, a vertex t_{i1} ; see Figure 1 in Appendix I [16] for an example of the full construction, combining the

single steps described in the original paper [6]. It comes as no surprise that this reduction does not work for us since, for example, $G_\Phi - v_s$ is always 3-colorable, and thus G_Φ is never stable if Φ is not satisfiable. However, careful checking reveals the following property of G_Φ .

► **Lemma 15.** *A 3CNF-formula Φ is not stable if and only if $\chi(G_\Phi) > \chi(G_\Phi - t_{i1})$ for at least one $i \in \{1, \dots, m\}$.*

The proof of Lemma 15 is deferred to Appendix H [16]. What we need now is a way to enhance the construction such that the deletion of a vertex other than t_{11}, \dots, t_{m1} , for example v_s , does not decrease the chromatic number. We achieve this by the following lemma.

► **Lemma 16.** *Let G be a graph and $v \in V(G)$. Let \widehat{G} be the graph that results from replicating v ; that is, $V(\widehat{G}) = V(G) \cup \{v'\}$ and $E(\widehat{G}) = E(G) \cup \{\{v', w\} \mid \{v, w\} \in E(G)\}$. Then $\chi(G) = \chi(\widehat{G}) = \chi(\widehat{G} - v) = \chi(\widehat{G} - v')$.*

Proof. The only nontrivial part is to show that $\chi(\widehat{G}) \leq \chi(G)$. To see this, we start with an arbitrary optimal valid vertex coloring of G and then color v' with the same color as v . ◀

Lemma 16 is simple and yet very powerful in our context. It allows us to select a set of vertices whose removal will not influence the chromatic number, and thus will not influence whether or not the graph is vertex-stable. We can use this to obtain the desired reduction.

► **Theorem 17.** *STABLE3CNF polynomial-time many-one reduces to VERTEXSTABILITY.*

Proof. Given a 3CNF-formula Φ , map it to $r(G_\Phi)$, where G_Φ is the graph from the reduction by Cai and Meyer [6] and r denotes the replication of all vertices other than t_{11}, \dots, t_{m1} .

If Φ is not in STABLE3CNF, then we have $\chi(G_\Phi) > \chi(G_\Phi - t_{i1})$ for some $i \in \{1, \dots, m\}$ by Lemma 15. Furthermore, a repeated application of Lemma 16 yields $\chi(r(G_\Phi)) = \chi(G_\Phi)$ and $\chi(r(G_\Phi) - t_{i1}) = \chi(r(G_\Phi - t_{i1})) = \chi(G_\Phi - t_{i1})$. Thus $r(G_\Phi)$ is not vertex-stable. For the converse, suppose that $r(G_\Phi)$ is not vertex-stable. Let $v \in V(r(G_\Phi))$ be a vertex such that $\chi(r(G_\Phi)) > \chi(r(G_\Phi) - v)$. From Lemma 16, we can see that $v = t_{i1}$ for some $i \in \{1, \dots, m\}$. By Lemma 15, this implies that Φ is not stable. ◀

This completes the proof of Theorem 8 – stating that VERTEXSTABILITY is Θ_2^P -complete – which in turn implies Theorem 7, the Θ_2^P -completeness of STABILITY, by Lemma 9. Now we briefly turn to some DP-complete problems. Recall that by prefixing a number k to the name of a stability property we additionally require the graph number to be exactly k .

► **Theorem 18.** *The problems k -STABILITY and k -VERTEXSTABILITY are NP-complete for $k = 3$ and DP-complete for $k \geq 4$.*

Proof. The membership proofs are immediate. For the lower bound we use that EXACT- k -COLORABILITY (the class of all graphs whose chromatic number is not merely at most, but exactly k) is NP-complete for $k = 3$ and DP-complete for $k \geq 4$; see [34]. It suffices to check that mapping G to $G \cup G$ reduces EXACT- k -COLORABILITY to k -STABILITY and k -VERTEXSTABILITY. Indeed, for any two graphs H and H' , we have $\chi(H \cup H') = \max\{\chi(H), \chi(H')\}$, implying that $G \cup G$ is stable and vertex-stable with $\chi(G) = \chi(G \cup G)$. ◀

In the previous proof, we used the disjoint union of a graph with itself to render it stable without changing its chromatic number. Using a far more complicated construction, we can also ensure the stability of an arbitrary set of edges of a graph while keeping track of how exactly this changes the chromatic number. We state this result in the following theorem.

19:10 Complexity of Stability

► **Lemma 19.** *There is a polynomial-time algorithm that, given any graph G plus a nonempty subset $S \subseteq E(G)$ of its edges, adds a fixed gadget to the graph and then substitutes for every $e \in S$ some gadget that depends on G and e , yielding a graph \widehat{G} with the following properties:*

1. $\chi(\widehat{G}) = \chi(G) + 2$.
2. All edges in $E(\widehat{G}) - (E(G) - S)$ are stable.
3. Each one of the remaining edges in $E(G) - S$ is stable in \widehat{G} exactly if it is stable in G .

Owing to space constraints, the proof of Lemma 19 is deferred to Appendix J [16]. However, we can at least provide a brief sketch of the construction for the case where only one edge is stabilized, that is $S = \{e\}$, omitting the verification of the properties. Join a new edge $\{w'_1, w'_2\}$ to the given G , remove e , join one of its endpoints to G'_e , an initially disjoint copy of G , and the other one to a new vertex, u'_e which is then in turn joined to G'_e . Finally replicate all vertices outside of G , yielding in particular an edge-free copy G''_e of G . Figure 2 in Appendix J [16] displays the relevant parts of the construction for a simple example with a singleton $S = \{e\}$.

Note that this construction allows us to reduce the problem of deciding whether in a given selection of edges all of them are stable to STABILITY by stabilizing all other edges. Moreover, it yields the following AND_ω function for STABILITY. This is stated in the following corollary, whose quite straightforward proof is deferred to Appendix K [16] due to the space constraints.

► **Corollary 20.** *Mapping k graphs G_1, \dots, G_k to $G_1 + \dots + G_k$ with all join edges stabilized using the construction from Lemma 19 is an AND_ω function for STABILITY.*

5 The Complexity of β -STABILITY and β -VERTEXSTABILITY

We will now examine the complexity of stability with respect to the vertex-cover number β .

First, note that β -VERTEXSTABILITY is trivially in P as it consists of the empty graphs.

► **Theorem 21.** *Only the empty graphs are β -vertex-stable.*

The easy proof is deferred to Appendix L [16]. Turning to the smaller change of deleting only an edge instead of a vertex, the situation changes radically. We will prove with Theorem 23 that determining whether a graph is β -stable is Θ_2^P -complete. An important ingredient to the proof is the following analogue to Lemma 16, which shows how to β -stabilize an arbitrary edge of a given graph. The proof is deferred to Appendix M [16] due to the space constraints.

► **Lemma 22.** *Let G be a graph and $\{v_1, v_2\} \in E(G)$ one of its edges. Create from G a new graph G' by replacing the edge $\{v_1, v_2\}$ by the gadget that consists of four new vertices u_1, u_2, u_3 , and u_4 with edges $\{u_1, u_2\}, \{u_2, u_3\}, \{u_3, u_4\}$, and $\{u_4, u_1\}$ (i.e., a new rectangle) and additionally the edges $\{v_1, u_1\}, \{v_1, u_3\}, \{v_2, u_2\}$, and $\{v_2, u_4\}$. (This gadget is displayed in Figure 3b in Appendix M [16].) Then we have $\beta(G') = \beta(G) + 2$, all edges of the gadget are stable in G' , and the remaining edges are stable in G' if and only if they are stable in G .*

► **Theorem 23.** *Determining whether a graph is β -stable is Θ_2^P -complete.*

Proof. We reduce from $\{(G, H) \in \mathcal{G}^2 \mid \beta(G) > \beta(H)\}$, which is Θ_2^P -hard [36, Thm. 12]. (Note that this language is essentially the complement of COMPAREVERTEXCOVER and that Θ_2^P is closed under taking the complement.) Let G and H be given graphs. Replace each edge $e \in E(G)$ by a copy of the stabilizing gadget described in Lemma 22. Call the resulting graph G' . Clearly, we have $\|V(G')\| = \|V(G)\| + 4\|E(G)\|$. By Lemma 22, G' is β -stable and $\beta(G') = \beta(G) + 2\|E(G)\|$. Moreover, let $H' = H \cup K_2$. The edge in K_2 ensures that H' is not β -stable. Moreover, we have $\beta(H') = \beta(H) + 1$ and $\|V(H')\| = \|V(H)\| + 2$.

Now, let $G'' = G'$, just for consistent notation, and $H'' = H' \cup K_{2\|E(G)\|}$. Since $\beta(K_n) = n - 1$ for $n \geq 1$, this implies $\beta(G'') - \beta(G) = \|E(G)\| = \beta(H'') - \beta(H)$. We finish the construction by adding isolated vertices to either G'' or H'' such that we achieve an equal number of vertices without changing the vertex cover number; that is, we let $G''' = G'' \cup I_{\max\{0, \|V(H'')\| - \|V(G'')\|\}}$ and $H''' = H'' \cup I_{\max\{0, \|V(G'')\| - \|V(H'')\|\}}$. Let $c = \|V(G''')\| = \|V(H''')\|$ and $d = \beta(G''') - \beta(G) = \beta(H''') - \beta(H)$. Note that G''' is β -stable since we stabilized G' with the gadget substitutions and then only added isolated vertices but no more edges. Moreover, H''' is not β -stable due to the β -critical edge of K_2 .

Let S be the join $G''' + H'''$ with all join edges stabilized, again by the gadget substitution described in Lemma 22. It is easy to see from the proof of Lemma 22 that the gadget as a whole behaves just like the edge it replaces, in the sense that an optimal vertex cover of the whole graph contains, without loss of generality, either v_1 or v_2 or both. Therefore, an optimal vertex cover of S consists of either an optimal vertex cover of G''' and all vertices of H''' or of an optimal vertex cover of H''' and all vertices of G''' plus, in both cases, a constant number k of vertices for covering the gadget edges – namely two for each former join edge, that is, $k = 2 \cdot \|V(G''')\| \cdot \|V(H''')\|$. In the first case, we obtain an optimal vertex cover for S of size $\beta(G''') = \beta(G) + d + c + k$, in the second case one of size $\beta(H''') = \beta(H) + d + c + k$.

Assume first that $\beta(G) > \beta(H)$. It follows that $\beta(G''') > \beta(H''')$ and thus any optimal vertex cover for S consists of all vertices $V(H''')$, an optimal vertex cover for G''' , and k vertices for the gadgets. Since we ensured that G''' is β -stable, S is β -stable. Now, assume that $\beta(G) \leq \beta(H)$. Then there is an optimal vertex cover that consists of all vertices of G''' , an optimal vertex cover of H''' , and again k vertices due to the gadgets. Since H''' not β -stable, as pointed out above, S is not β -stable either. We conclude that S is β -stable exactly if $\beta(G) > \beta(H)$, thus proving that β -stability is Θ_2^P -hard and therefore Θ_2^P -complete. ◀

6 Unfrozenness

We begin with the observation that both for Colorability and for Vertex Cover adding a vertex is too generous a modification to be interesting. The trivial proof is found in Appendix N [16].

► **Theorem 24.** *There is no vertex-unfrozen graph and only one β -vertex-unfrozen graph, namely the null graph (i.e., the graph with the empty vertex set).*

Both problems are far more interesting in the default setting, that is, for adding edges. The Θ_2^P -completeness of deciding whether a given graph is β -unfrozen can be obtained by a method similar to the one we used to establish Theorem 23; see Appendix O [16] for the proof.

► **Theorem 25.** *Determining whether a graph is β -unfrozen is Θ_2^P -complete.*

Now, we would like to show the analogous result that UNFROZENNESS is Θ_2^P -complete as well. This turns out to be a very difficult task, however. There are many clues suggesting the hardness of UNFROZENNESS, which exhibits a far richer structure than all of the problems listed in Table 1 as easy. The latter problems are either empty or singletons or consist of all independent sets or all cliques, while UNFROZENNESS contains large classes of different graphs. We can even produce arbitrarily many new complicated unfrozen graphs using the graph join. There are no clearly identifiable characteristics to these unfrozen graphs to be leveraged. Instead, we give a sufficient condition for the Θ_2^P -completeness of UNFROZENNESS, namely the existence of a polynomial-time computable construction that turns arbitrary graphs into unfrozen ones without changing their chromatic number in an intractable way.

19:12 Complexity of Stability

► **Theorem 26.** *Assume that there are polynomial-time computable functions $f: \mathcal{G} \rightarrow \mathcal{G}$ and $g: \mathcal{G} \rightarrow \mathbb{Z}$ such that for any graph G we have that $f(G)$ is unfrozen and $\chi(f(G)) = \chi(G) + g(G)$. Then UNFROZENNESS is Θ_2^P -complete.*

The proof of Theorem 26 is deferred to Appendix P [16]. It is similar in flavor to the one of Theorem 25 and reduces from COMPARECOLORABILITY, whose Θ_2^P -hardness is stated now.

► **Theorem 27.** *COMPARECOLORABILITY = $\{(G, H) \in \mathcal{G}^2 \mid \chi(G) \leq \chi(H)\}$ is Θ_2^P -hard.*

Theorem 27 is proved essentially in the same way as Wagner [38, Thm. 6.3.2] proves the Θ_2^P -hardness of ODDCOLORABILITY. As he suggests [38, page 79], it is rather straightforward to translate the hardness result for ODDCOLORABILITY into one for EQUALCOLORABILITY. This holds true for COMPARECOLORABILITY as well. The method for obtaining these results is easily generalized to yield two sufficient criteria for Θ_2^P -hardness, stated as Lemmas 33 and 34 in Appendix Q [16]. We use the latter lemma – stated in a somewhat flawed form by Spakowski and Vogel [36, Lem. 9] – to prove Theorem 27. See Appendix Q [16] for all details.

Note that an analogue to Lemma 19 for unfreezing instead of stabilizing edges would be sufficient to satisfy the assumption of Theorem 26. However, based on our efforts we suspect that a suitable gadget – if one exists – must be of significantly higher complexity than the one in Figure 2 in Appendix J [16].

7 Two-Way Stability

A graph is two-way stable if it is stable with respect to both the deletion and addition of an edge. First, we note that the analogous problem with respect to vertices is trivial for both Colorability and Vertex Cover. The following is an immediate consequence of Theorem 24.

► **Theorem 28.** *There is no vertex-two-way-stable graph and only one β -vertex-two-way-stable graph, namely the null graph with the empty vertex set.*

The default case of edge deletion is more interesting. We begin with Colorability.

► **Theorem 29.** *The problem TWOWAYSTABILITY is Θ_2^P -complete.*

To prove this, it suffices to check that mapping a graph G to $G \cup G$ reduces UNFROZENNESS to TWOWAYSTABILITY; see Appendix R [16] for the details. We are able to prove the analogous result for β -TWOWAYSTABILITY via Lemma 30; the proof is deferred to Appendix S [16].

► **Lemma 30.** *Let a nonempty graph G and an edge $e \in E(G)$ be given. Construct from G a graph G' by substituting for e the constant-size gadget that consists of a clique on the new vertex set $\{u_1, u_2, u_3, u_4, u'_1, u'_2, u'_3, u'_4\}$, with the four edges $\{u_i, u'_i\}$ for $i \in \{1, 2, 3, 4\}$ removed and the four edges $\{v, u_1\}$, $\{v, u_2\}$, $\{v', u_3\}$, and $\{v', u_4\}$ added. (This gadget is displayed in Figure 5b in Appendix S [16].) The graph G' has the following properties.*

1. $\beta(G') = \beta(G) + 6$,
2. every edge $e' \in E(G) - \{e\}$ is β -stable in G exactly if it is in G' ,
3. all remaining edges of G' are β -stable,
4. every nonedge $e' \in \overline{E}(G)$ is β -unfrozen in G exactly if it is in G' , and
5. all remaining nonedges $e' \in \overline{E}(G') - \overline{E}(G)$ of G' are β -unfrozen.

An iterated application of this lemma allows us to stabilize an arbitrary set of edges of an arbitrary graph without introducing any new unfrozen edges. The Θ_2^P -hardness of β -TWOWAYSTABILITY is now an easy consequence of Lemma 30; see Appendix T [16].

► **Theorem 31.** *The problem β -TWOWAYSTABILITY is Θ_2^P -complete.*

References

- 1 Douglas Bauer, Frank Harary, Juhani Nieminen, and Charles L. Suffel. Domination alteration sets in graphs. *Discrete Mathematics*, 47:153–161, 1983.
- 2 Adam Beacham and Joseph C. Culberson. On the complexity of unfrozen problems. *Discrete Applied Mathematics*, 153(1-3):3–24, 2005.
- 3 Béla Bollobás. *Modern Graph Theory*. Springer-Verlag, 1998.
- 4 Béla Bollobás. *Extremal Graph Theory*. London Mathematical Society Monographs. Oxford University Press, London, 1978.
- 5 Elisabet Burjons, Fabian Frei, Edith Hemaspaandra, Dennis Komm, and David Wehner. Finding optimal solutions with neighborly help. In *Proceedings of the 44th International Symposium on Mathematical Foundations of Computer Science (MFCS 2019)*, volume 138 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 78:1–78:14. Schloss Dagstuhl, 2019.
- 6 Jin-Yi Cai and Gabriele E. Meyer. Graph minimal uncolorability is D^P -complete. *SIAM Journal on Computing*, 16(2):259–277, 1987.
- 7 Gregory J. Chaitin. Register allocation and spilling via graph coloring. *ACM SIGPLAN Notices – Proceedings of the 1982 SIGPLAN Symposium on Compiler Construction*, 17(6):98–101, 1982.
- 8 R. Chang and J. Kadin. The Boolean hierarchy and the polynomial hierarchy: A closer connection. *SIAM Journal on Computing*, 25(2):340–354, 1996.
- 9 Richard Chang and Jim Kadin. On computing boolean connectives of characteristic functions. *Mathematical Systems Theory*, 28(3):173–198, 1995.
- 10 Kamalika Chaudhuri, Fan Chung, and Mohammad Sh. Jamall. A network coloring game. In *Proceedings of the 4th International Workshop On Internet And Network Economics*, pages 522–530. Springer-Verlag *Lecture Notes in Computer Science #5385*, December 2008.
- 11 Guantao Chen and Guangming Jing. Structural properties of edge-chromatic critical multi-graphs. *Journal of Combinatorial Theory, Series B*, 139:128–162, 2019.
- 12 Wyatt J. Desormeaux, Teresa W. Haynes, and Michael A. Henning. Total domination critical and stable graphs upon edge removal. *Discrete Applied Mathematics*, 158(15):1587–1592, 2010.
- 13 Gabriel A. Dirac. Some theorems on abstract graphs. *Proceedings of the London Mathematical Society*, s3-2(1):69–81, 1952.
- 14 Paul Erdős and Tibor Gallai. On the minimal number of vertices representing the edges of a graph. *Publications of the Mathematical Institute of the Hungarian Academy of Sciences*, 6(1-2):181–203, 1961.
- 15 Martin G. Everett and Steve Borgatti. Role colouring a graph. *Mathematical Social Sciences*, 21(2):183–188, 1991.
- 16 Fabian Frei, Edith Hemaspaandra, and Jörg Rothe. Complexity of stability. Technical Report arXiv:1910.00305 [cs.CC], arXiv.org, September 2020. [arXiv:1910.00305](https://arxiv.org/abs/1910.00305).
- 17 Georg Gunther, Bert Hartnell, and Douglas F. Rall. Graphs whose vertex independence number is unaffected by single edge addition or deletion. *Discrete Applied Mathematics*, 46:167–172, 1993.
- 18 Frank Harary. *Graph Theory*. Addison-Wesley, 1969.
- 19 Frank Harary and Carsten Thomassen. Anticritical graphs. *Mathematical Proceedings of the Cambridge Philosophical Society*, 79:11–18, 1976.
- 20 Teresa W. Haynes, Robert C. Brigham, and Ronald D. Dutton. Extremal graphs domination insensitive to the removal of k edges. *Discrete Applied Mathematics*, 44(1-3):295–304, 1993.
- 21 Lane A. Hemaspaandra. The strong exponential hierarchy collapses. *Journal of Computer and System Sciences*, 39(3):299–322, 1989.
- 22 Edith Hemaspaandra, Lane A. Hemaspaandra, and Jörg Rothe. Exact analysis of Dodgson elections: Lewis Carroll’s 1876 voting system is complete for parallel access to NP. *Journal of the ACM*, 44(6):806–825, 1997.

- 23 Edith Hemaspaandra, Lane A. Hemaspaandra, and Jörg Rothe. The complexity of online manipulation of sequential elections. *Journal of Computer and System Sciences*, 80(4):697–710, 2014.
- 24 Edith Hemaspaandra, Holger Spakowski, and Jörg Vogel. The complexity of Kemeny elections. *Theoretical Computer Science*, 349(3):382–391, 2005.
- 25 Michael A. Henning and Marcin Krzywkowski. Total domination stability in graphs. *Discrete Applied Mathematics*, 236:246–255, 2018.
- 26 Matthew O. Jackson. *Social and Economic Networks*. Princeton University Press, 2008.
- 27 Michael Kearns, Siddharth Suri, and Nick Montfort. An experimental study of the coloring problem on human subject networks. *Science*, 313(5788):824–827, 2006.
- 28 Susan Khor. Application of graph coloring to biological networks. *IET Systems Biology*, 4(3):185–192, 2010.
- 29 Frank Th. Leighton. A graph coloring algorithm for large scheduling problems. *Journal of Research of the National Bureau of Standards*, 84(6):489–506, 1979.
- 30 Rémi Monassen and Riccardo Zecchina. Statistical mechanics of the random k -satisfiability model. *The American Physical Society*, 56(2):1357–1370, 1997.
- 31 Rémi Monassen, Riccardo Zecchina, Scott Kirkpatrick, Bart Selman, and Lidror Troyansky. Determining computational complexity from characteristic ‘phase transitions’. *Nature*, 400:133–137, 1998.
- 32 Christos H. Papadimitriou and David Wolfe. The complexity of facets resolved. *Journal of Computer and System Sciences*, 37(1):2–13, 1988.
- 33 Christos H. Papadimitriou and Mihalis Yannakakis. The complexity of facets (and some facets of complexity). *Journal of Computer and System Sciences*, 28(2):244–259, 1984.
- 34 Jörg Rothe. Exact complexity of Exact-Four-Colorability. *Information Processing Letters*, 87(1):7–12, 2003.
- 35 Jörg Rothe, Holger Spakowski, and Jörg Vogel. Exact complexity of the winner problem for Young elections. *Theory of Computing Systems*, 36(4):375–386, 2003.
- 36 H. Spakowski and J. Vogel. Θ_2^p -completeness: A classical approach for new results. In *Proceedings of the 20th Conference on Foundations of Software Technology and Theoretical Computer Science*, volume 1974 of *Lecture Notes in Computer Science*, pages 348–360. Springer-Verlag, December 2000.
- 37 Jimeng Sun, Charalampos E. Tsourakakis, Evan Hoke, Christos Faloutsos, and Tina Eliassi-Rad. Two heads better than one: Pattern discovery in time-evolving multi-aspect data. *Data Mining and Knowledge Discovery*, 17(1):111–128, 2008.
- 38 Klaus W. Wagner. More complicated questions about maxima and minima, and some closures of NP. *Theoretical Computer Science*, 51(1–2):53–80, 1987.
- 39 Klaus W. Wagner. Bounded query classes. *SIAM Journal on Computing*, 19(5):833–846, 1990.
- 40 Walter Wessel. Criticity with respect to properties and operations in graph theory. In László Lovász András Hajnal and Vera T. Sós, editors, *Finite and Infinite Sets. (6th Hungarian Combinatorial Colloquium, Eger, 1981)*, volume 2 of *Colloquia Mathematica Societatis Janos Bolyai*, pages 829–837. North-Holland, 1984.
- 41 Gerhard J. Woeginger. Core stability in hedonic coalition formation. In *Proceedings of the 39th International Conference on Current Trends in Theory and Practice of Computer Science*, volume 7741 of *Lecture Notes in Computer Science*, pages 33–50. Springer-Verlag, 2013.