Unbounded-Width CSPs Are Untestable in a Sublinear Number of Queries

Yumou Fei
Massachusetts Institute of Technology, Cambridge, MA, USA

Abstract

The bounded-degree query model, introduced by Goldreich and Ron (*Algorithmica, 2002*), is a standard framework in graph property testing and sublinear-time algorithms. Many properties studied in this model, such as bipartiteness and 3-colorability of graphs, can be expressed as satisfiability of constraint satisfaction problems (CSPs). We prove that for the entire class of *unbounded-width* CSPs, testing satisfiability requires $\Omega(n)$ queries in the bounded-degree model. This result unifies and generalizes several previous lower bounds. In particular, it applies to all CSPs that are known to be **NP**-hard to solve, including k-colorability of ℓ-uniform hypergraphs for any $k, \ell \geqslant 2$ with $(k, \ell) \neq (2, 2)$.

Our proof combines the techniques from Bogdanov, Obata, and Trevisan (*FOCS, 2002*), who established the first $\Omega(n)$ query lower bound for CSP testing in the bounded-degree model, with known results from universal algebra.

2012 ACM Subject Classification Theory of computation → Computational complexity and cryptography

Keywords and phrases constraint satisfaction problems, property testing

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2026.31

Category RANDOM

Related Version *Full Version*: `https://arxiv.org/abs/2510.27012`

Acknowledgements The author would like to thank Dor Minzer, Ronitt Rubinfeld, Shuo Wang and Standa Živný for many stimulating discussions during the development of this work. Special thanks to Ronitt Rubinfeld, for her helpful suggestions in improving the presentation of the paper.

1 Introduction

Property testing is a class of algorithmic problems in which the algorithm is allowed to inspect only a small portion of the input. In exchange for this limited access, the goal is relaxed: instead of deciding whether the input exactly satisfies a given property, the tester must merely distinguish inputs that satisfy the property from those that are *far* from satisfying it, under an appropriate notion of distance.

In this paper, we consider the problem of testing satisfiability of constraint satisfaction problems (CSPs). We formally define CSPs as follows. First of all, every CSP considered in this paper is specified by a finite *template*.

▶ **Definition 1.** *A* relational structure *is a pair (D, Γ) where D is a finite set, and Γ is a finite collection of relations on D. Each relation $R \in \Gamma$ is a function $R : D^k \to \{0, 1\}$ for some positive integer k (called the* arity *of R). A fixed relational structure (D, Γ) is also referred to as a* CSP template.

As a running example, if we let $D = \{0, 1\}$ and take $\Gamma = \{\mathbb{1}_{\{10,01\}}\}$, where $\mathbb{1}_{\{10,01\}}(x, y) = 1$ if and only if $x \neq y$, then the template (D, Γ) could express the bipartiteness property of graphs, as we will soon explain.

Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2026).
Editors: Mohit Singh and Tom Gur; Article No. 31; pp. 31:1–31:9
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Given a CSP template (D, Γ), we can collect all *instances* of the CSP into an infinite set $\mathrm{CSP}(\Gamma)$.

► **Definition 2.** *Given a CSP template* (D, Γ)*, we define* $\mathrm{CSP}(\Gamma)$ *as the (infinite) collection of all* instances $I = (V, \mathcal{C})$*, where* V *is a finite set of* variables *and* $\mathcal{C} = \{C_1, \ldots, C_m\}$ *is a multi-set of* constraints *satisfying the following. Each constraint* C_i *is a pair* $(\mathbf{v}, R)$*, where* $\mathbf{v} = (v_1, \ldots, v_k)$ *is a tuple of distinct*[1] *variables from* V*, and* $R : D^k \to \{0, 1\}$ *is a relation from* Γ*. The variable set* $\{v_1, \ldots, v_k\} \subseteq V$ *is called the* scope *of* C_i*.*

Now an instance in $\mathrm{CSP}(\{\mathbb{1}_{\{10,01\}}\})$ can be viewed as a graph, with variables as vertices and constraints as edges. The graph is bipartite if and only if the CSP instance is "satisfiable," as defined in the following definition.

► **Definition 3.** *An* assignment *for an instance* $I = (V, \mathcal{C}) \in \mathrm{CSP}(\Gamma)$ *is a map* $\tau : V \to D$*. We say* τ satisfies *a constraint* $((v_1, \ldots, v_k), R)$ *if* $R\big(\tau(v_1), \ldots, \tau(v_k)\big) = 1$*. The* value *of an assignment* τ*, denoted by* $\mathrm{val}_I(\tau)$*, is the fraction of constraints in* $\mathcal{C}$ *satisfied by* τ*. Note that* $\mathrm{val}_I(\tau) \in [0, 1]$*. If* $\mathrm{val}_I(\tau) = 1$*, we say that* τ satisfies *the instance* I*, or* τ *is a* satisfying assignment *of* I*. The* value *of the instance* I*, denoted by* val_I*, is the maximum possible value any assignment may achieve.*

The CSP defined by the bipartiteness template $\big(\{0, 1\}, \{\mathbb{1}_{\{10,01\}}\}\big)$ will be denoted by 2COL. Similarly, the property of vertex 3-colorability in graphs can be expressed as a constraint satisfaction problem denoted by 3COL.

1.1 Testing CSP Satisfiability

We study the property testing of CSP satisfiability in the *bounded-degree query model*, defined in Definition 5 below. In this model, the input CSP instance has a fixed variable set, and the degree of each variable (i.e. the number of constraints involving the variable) in the instance is assumed to be uniformly bounded by a constant.

► **Definition 4.** *Let* (D, Γ) *be a CSP template and let* V *be a set of variables. We write* $\mathrm{CSP}(\Gamma, V) \subseteq \mathrm{CSP}(\Gamma)$ *for the collection of all* $\mathrm{CSP}(\Gamma)$*-instances whose variable set is* V*.*

► **Definition 5.** *Fix a CSP template* (D, Γ)*. In the* $\mathrm{BD}(d, n)$ *model for testing satisfiability, the tester is given oracle access to a CSP instance* $I = ([n], \mathcal{C}) \in \mathrm{CSP}(\Gamma, [n])$ *with maximum degree at most* d*. The goal of an* ε*-tester, where* $\varepsilon > 0$ *is a fixed constant, is to achieve the following:*

1. *If* I *is satisfiable, the tester must accept* I *with probability at least* $2/3$*.*
2. *If at least* εdn *constraints must be removed from* $\mathcal{C}$ *to make* I *satisfiable, the tester must reject* I *with probability at least* $2/3$*.*

The tester accesses the instance I *through* oracle queries*: when the tester queries a variable* $v \in [n]$*, the oracle reveals an arbitrary one of the previously unseen constraints involving* v*, unless all constraints involving* v *have been revealed (in which case the oracle returns* $\perp$*).*

The bounded-degree query model was first introduced by [17], in the context of graph property testing. One of the earliest results in this model is that testing bipartiteness (i.e. testing satisfiability of 2COL instances) requires at least $\Omega(\sqrt{n})$ queries [17] (for some degree

[1] Some authors allow repetition of variables in a constraint. In our setting, one can also "artificially" allow repetitions by adding more relations to the relational structure. See the full version of this paper for details.

bound d and error $\varepsilon > 0$) and can be done with at most $\widetilde{O}(\sqrt{n})$ queries [16] (for any degree bound d and any error $\varepsilon > 0$). In contrast, Bogdanov, Obata, and Trevisan [6] showed that testing vertex 3-colorability requires $\Omega(n)$ queries.[2]

The study of testing CSP satisfiability in the bounded-degree model was already formally initiated in the early work of Bogdanov, Obata, and Trevisan [6], who showed that, beyond the specific case of 3COL, several other CSPs (such as 3SAT and 3LIN) require $\Omega(n)$ queries to test in this model. A more systematic investigation was later carried out by Yoshida [27], who proved that any CSP *robustly solved by BasicLP* (the basic linear program) has a satisfiability tester that makes a constant number of queries. Typical examples of such CSPs include 1SAT and 3HornSAT. For the satisfiability testing of any CSP that is not *solved by BasicLP*, [27] further showed that at least $\Omega(\sqrt{n})$ queries are necessary. Examples of such CSPs include 2SAT and 2COL.[3]

The main result of this paper strengthens this picture by proving that, among these BasicLP-unsolvable CSPs, the subclass of *unbounded-width* CSPs requires $\Omega(n)$ queries to test.

▶ **Theorem 6.** *For any CSP template (D, Γ) of unbounded width, there exist constants $\varepsilon \in (0, 1)$ and $d \in \mathbb{N}$ such that any ε-tester for satisfiability of $\mathrm{CSP}(\Gamma)$ instances in the $\mathrm{BD}(d, n)$ model must make $\Omega_{d,\varepsilon}(n)$ queries.*

The notion of bounded width (formally defined in Appendix A) originates from [12] and has been extensively studied in the universal-algebraic approach to CSPs. Bounded-width CSPs can be solved in polynomial time (see Proposition 15), so the class of unbounded-width CSPs includes all CSPs whose satisfiability is **NP**-hard to decide (unless **P** = **NP**, in which case all CSPs are trivially **NP**-hard). Consequently, Theorem 6 implies that all CSPs known to be **NP**-hard are also unconditionally (and maximally) hard to test in the bounded-degree model. This encompasses, for example, the property of k-colorability in ℓ-uniform hypergraphs for all $k, \ell \geqslant 2$ with $(k, \ell) \neq (2, 2)$ (see Remark A.5 of the full version). Our result thus unifies and extends several prior lower bounds from works such as [6, 28, 1].[4]

1.2 Discussion: Testing vs. Approximation

A property tester can be viewed as an algorithm that approximates the distance of an input to satisfying a property – albeit in a very coarse sense, as it only distinguishes zero distance from sufficiently large distance. Nevertheless, many property testers (for example, the constant-query tester of [27] for certain CSPs) operate by explicitly estimating this distance and checking whether it is close to zero. On the negative side, a hardness result for property testing (such as our Theorem 6) can often be interpreted as a hardness of approximation result. This connection to approximation algorithms was, in fact, one of the original motivations in [15], where graph property testing was first introduced.

In this subsection, we examine and discuss our main result Theorem 6 from the perspective of sublinear-time approximation algorithms.

---

[2] These seminal works, along with a large number of subsequent studies (such as [18, 5, 28, 29, 10], to name a few), also investigated a wide range of other graph properties within the bounded-degree model, many of which are not constraint satisfaction problems.

[3] Robust solvability by BasicLP is a stronger condition than solvability by BasicLP, so [27] left open the possibility of CSPs with superconstant but $o(\sqrt{n})$ query complexity. However, a recent unpublished note by Brady [7] proves that all CSPs solvable by BasicLP are also robustly solvable by it, thereby turning the result of [27] into a full dichotomy theorem.

[4] The recent work of [1] proves that testing k-colorability in k-uniform hypergraphs requires $\Omega(n)$ queries, for any $k \geqslant 3$.

1.2.1 MaxCSP Problems

A natural approximation problem associated with CSPs is that of estimating the value of a CSP instance (as defined in Definition 3).[5] This estimation task can be conveniently formulated in the following "gap version."

▶ **Definition 7.** *For a fixed CSP template (D, Γ), a completeness parameter $c \in (0, 1]$ and a soundness parameter $s \in [0, c)$, the problem* $\mathsf{MaxCSP}(\Gamma)[c, s]$ *is the promise problem where given an instance $I \in \mathrm{CSP}(\Gamma)$, the (randomized) algorithm should distinguish between the following two cases:*

1. *Yes case: if $\mathrm{val}_I \geqslant c$, then the algorithm should accept with probability at least $2/3$.*
2. *No case: if $\mathrm{val}_I \leqslant s$, then the algorithm should reject with probability at least $2/3$.*

The complexity of MaxCSP problems has been extensively studied in the context of polynomial-time algorithms, and a central question is to determine for which tuples (Γ, c, s) the problem $\mathsf{MaxCSP}(\Gamma)[c, s]$ can be solved in polynomial time. Notably, Raghavendra [24] almost[6] characterizes all such tuples, conditioned on the Unique Games Conjecture of [23].

1.2.2 MaxCSP vs. Satisfiability Testing

Whether property testing of CSP satisfiability has implications for MaxCSP problems depends on the notion of "distance to satisfiability" used by the property tester. In the d-bounded-degree model, having distance ε to satisfiability means that at least εdn constraints must be removed to make the instance satisfiable. In contrast, an instance with value $1 - \varepsilon$ in the MaxCSP sense requires removing at least an ε *fraction* of its constraints. Thus, testing CSP satisfiability in the bounded-degree model is analogous to solving $\mathsf{MaxCSP}(\Gamma)[1, 1 - \varepsilon]$, provided the CSP instance has $\Theta(n)$ constraints.

However, a CSP instance in the d-bounded-degree model $\mathrm{BD}(d, n)$ may have far fewer than $\Theta(n)$ constraints. In fact, if an instance contains only $O(1)$ constraints, it would be unreasonable to expect any sublinear-time algorithm to approximate its value, since with high probability the queried variables would not appear in any constraint. To address this, we introduce a modified model $\mathrm{BD}^*(d, \alpha, n)$, where $\alpha > 0$ is a fixed constant ensuring that all CSP instances considered contain at least αn constraints.

Under this refinement, Theorem 6 can be equivalently stated as follows:

▶ **Theorem 8.** *For any CSP template (D, Γ) of unbounded width, there exist constants $\varepsilon, \alpha \in (0, 1)$ and $d \in \mathbb{N}$ such that any algorithm for* $\mathsf{MaxCSP}(\Gamma)[1, 1 - \varepsilon]$ *in the* $\mathrm{BD}^*(d, \alpha, n)$ *model must make $\Omega(n)$ queries.*

It is easy to see that Theorem 8 implies Theorem 6. Due to the convenience provided by this rephrased version, in the rest of the paper we will work with Theorem 8 instead of Theorem 6.

---

[5] Ideally, an approximation algorithm would also produce an assignment achieving this value, i.e. one that satisfies as many constraints as possible. However, since we focus on sublinear-time algorithms in this paper, we cannot expect them to explicitly output a complete assignment.

[6] Except for the perfect-completeness case (i.e. the case $c = 1$), which remains a mystery.

1.2.3 Sublinear Time vs. Polynomial Time

As discussed in Section 1.2.2, because of the specific notion of "distance to satisfiability" used in the bounded-degree model, satisfiability testing in this setting most naturally corresponds to MaxCSP problems on instances with $\Theta(n)$ constraints. In contrast, the *dense graph model* (the more extensively studied model in graph property testing) uses another notion of distance, making it better aligned with MaxCSP problems on "dense" instances.

We argue that exactly due to this difference in instance regimes, the bounded-degree model provides a stronger connection to the study of polynomial-time algorithms for MaxCSPs, at least with respect to worst-case approximation ratios. First, MaxCSP problems on general instances can often be reduced to bounded-degree instances (see, e.g., [26]), so bounded-degree instances already capture much of the hardness in the problems. Second, MaxCSP problems tend to be easier on dense instances (see, e.g., [3]). Indeed, [2] shows that every CSP admits a constant-query satisfiability tester in the dense graph model. In contrast, in the bounded-degree model, satisfiability testing already exhibits a wide spectrum of complexities across CSP templates, with query complexity ranging from constant, to $\widetilde{\Theta}(\sqrt{n})$, up to $\Theta(n)$.

We next discuss some results about MaxCSP in the polynomial-time setting that are most relevant to this work.

In the context of polynomial-time algorithms, for certain CSP templates (D, Γ), the *perfect-completeness* problem $\mathsf{MaxCSP}(\Gamma)[1, 1-\varepsilon]$ can exhibit a markedly different level of complexity from the (potentially harder) *imperfect-completeness* problem $\mathsf{MaxCSP}(\Gamma)[1-\varepsilon', 1-\varepsilon]$, even when $\varepsilon' \to 0$ and ε is fixed. For example, in the case of 3LIN (an unbounded-width CSP), the former problem admits a polynomial-time algorithm via Gaussian elimination, whereas the latter is **NP**-hard (due to [20]). More generally, Dalmau and Krokhin [11] proved that the imperfect-completeness problem is **NP**-hard for all unbounded-width CSP templates.

▶ **Theorem 9** ([11]). *For any CSP template (D, Γ) of unbounded width, there exists a constant $\varepsilon \in (0, 1)$ such that for every constant $\varepsilon' \in (0, \varepsilon)$, the problem $\mathsf{MaxCSP}(\Gamma)[1-\varepsilon', 1-\varepsilon]$ is* **NP***-hard.*

In terms of conclusions, our result is not directly comparable with Theorem 9, for multiple reasons. First, Theorem 9 is a hardness result for the imperfect completeness problem, while our Theorem 8 shows hardness for the (potentially easier) perfect completeness problem. Second, Theorem 9 is proved against polynomial-time algorithms, while Theorem 8 is against sublinear-query algorithms. Although **NP**-hardness is likely much stronger than a linear query lower bound, the latter does have the benefit of being unconditional. It is also not clear a priori whether some useful algorithms could have sublinear query complexity but super-polynomial time complexity, so **NP**-hardness usually does not immediately imply query lower bounds in sublinear-time models, even when assuming $\mathbf{P} \neq \mathbf{NP}$.

In terms of techniques, however, the proof of Theorem 8 runs in direct parallel to that of Theorem 9 in [11]. While [11] constructs a reduction from the **NP**-hardness of approximating 3LIN instances (established by [20]), our proof of Theorem 8 builds an analogous reduction from the linear-query lower bound for 3LIN instances (proved in [6]). Unlike in the **NP** setting, where a reduction need only run in polynomial time, our bounded-degree query setting requires addressing several additional subtleties. In particular, certain parts of the proof rely on a careful combination of techniques from both [6] and [11].

▶ Remark 10. The result of [11] (Theorem 9) was later shown to be tight by Barto and Kozik [4]. Specifically, they proved that for every CSP template (D, Γ) of bounded width and every constant $\varepsilon \in (0, 1)$, there exists a constant $\varepsilon' \in (0, \varepsilon)$ such that the problem

$\mathsf{MaxCSP}(\Gamma)[1-\varepsilon', 1-\varepsilon]$ belongs to $\mathbf{P}$. Together with the result of [11], this establishes that bounded width exactly characterizes the class of *robustly solvable* CSPs, as conjectured earlier by Guruswami and Zhou [19].

1.2.4 Sublinear Time vs. Sublinear Space

The past decades have seen a growing body of work on MaxCSP problems in *sublinear-space* algorithmic models (such as [21, 22, 9, 8, 25, 13], to name a few), particularly in the contexts of sketching and streaming algorithms. Recent studies [14, 13] reveal a phenomenological connection between the *multi-pass streaming* model and the bounded-degree query model: the class of MaxCSP problems that are easy in the bounded-degree model (admitting algorithms with constantly many queries) almost[7] coincides with those that are easy in the multi-pass streaming model (admitting algorithms with constantly many passes and polylogarithmic space). We view this connection as further motivation for studying MaxCSP and other approximation problems in the bounded-degree query model.

In light of this connection, a natural question arises: does the hardness result of Theorem 8 also extend to the multi-pass streaming model?

▶ **Question 11.** *Is it true that for every CSP template (D, Γ) of unbounded width, there exists a constant $\varepsilon > 0$ such that $\mathsf{MaxCSP}(\Gamma)[1, 1-\varepsilon]$ requires $\Omega(n)$ space in the multi-pass streaming model?*

We remark that, as discussed in [13], query lower bounds in sublinear time models are usually easier to establish than space lower bounds, and thus Theorem 8 can be seen as a preliminary step towards answering Question 11.

1.3 Open Problems

Perhaps the most obvious open problem left by this work is whether the result of Theorem 8 is tight, analogous to the case of Theorem 9 (see Remark 10).

▶ **Question 12.** *Do sublinear-query algorithms exist for satisfiability testing of all bounded-width CSPs?*

In their original work, Bogdanov, Obata and Trevisan [6] asked whether, given a CSP template (D, Γ), one can determine the optimal hardness gap ε for which $\mathsf{MaxCSP}(\Gamma)[1, 1-\varepsilon]$ requires $\Omega(n)$ queries. This question remains wide open (even for "simple" templates such as $\mathsf{3COL}$). Here, we pose the following more ambitious version.

▶ **Question 13.** *Determine the query complexity of $\mathsf{MaxCSP}(\Gamma)[c, s]$ in the bounded-degree model, for all predicate families Γ and parameters c and s.*

As discussed in Section 1.2.4, this question may also have strong relevance to sublinear-space computational models. Moreover, resolving Question 13 unconditionally (without assuming $\mathbf{P} \neq \mathbf{NP}$ or the Unique Games Conjecture) could possibly shed new light on the corresponding question for polynomial-time algorithms.

[7] Except for the perfect-completeness case, which remains a mystery for both models.

References

1. Hugo Aaronson, Gaia Carenini, and Atreyi Chanda. Property testing in bounded degree hypergraphs. *arXiv preprint arXiv:2502.18382*, 2025. doi:10.48550/arXiv.2502.18382.
2. Noga Alon and Asaf Shapira. Testing satisfiability. *Journal of Algorithms*, 47(2):87–103, 2003. doi:10.1016/S0196-6774(03)00019-1.
3. Sanjeev Arora, David Karger, and Marek Karpinski. Polynomial time approximation schemes for dense instances of np-hard problems. In *Proceedings of the twenty-seventh annual ACM symposium on Theory of computing*, pages 284–293, 1995. doi:10.1145/225058.225140.
4. Libor Barto and Marcin Kozik. Robustly solvable constraint satisfaction problems. *SIAM Journal on Computing*, 45(4):1646–1669, 2016. doi:10.1137/130915479.
5. Itai Benjamini, Oded Schramm, and Asaf Shapira. Every minor-closed property of sparse graphs is testable. *Advances in Mathematics*, 223:2200–2218, 2010.
6. Andrej Bogdanov, Kenji Obata, and Luca Trevisan. A lower bound for testing 3-colorability in bounded-degree graphs. In *The 43rd Annual IEEE Symposium on Foundations of Computer Science, 2002. Proceedings.*, pages 93–102. IEEE, 2002. doi:10.1109/SFCS.2002.1181886.
7. Zarathustra Brady. Notes on csps and polymorphisms. *arXiv preprint arXiv:2210.07383*, 2022. doi:10.48550/arXiv.2210.07383.
8. Chi-Ning Chou, Alexander Golovnev, Madhu Sudan, Ameya Velingker, and Santhoshini Velusamy. Linear space streaming lower bounds for approximating csps. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, pages 275–288, 2022. doi:10.1145/3519935.3519983.
9. Chi-Ning Chou, Alexander Golovnev, Madhu Sudan, and Santhoshini Velusamy. Sketching approximability of all finite csps. *Journal of the ACM*, 71(2):1–74, 2024. doi:10.1145/3649435.
10. Artur Czumaj, Morteza Monemizadeh, Krzysztof Onak, and Christian Sohler. Planar graphs: Random walks and bipartiteness testing. *Random Structures & Algorithms*, 55(1):104–124, 2019. doi:10.1002/RSA.20826.
11. Víctor Dalmau and Andrei Krokhin. Robust satisfiability for csps: Hardness and algorithmic results. *ACM Transactions on Computation Theory (TOCT)*, 5(4):1–25, 2013. doi:10.1145/2540090.
12. Tomás Feder and Moshe Y Vardi. The computational structure of monotone monadic snp and constraint satisfaction: A study through datalog and group theory. *SIAM Journal on Computing*, 28(1):57–104, 1998. doi:10.1137/S0097539794266766.
13. Yumou Fei, Dor Minzer, and Shuo Wang. A dichotomy theorem for multi-pass streaming csps. *arXiv preprint arXiv:2509.11399*, 2025. doi:10.48550/arXiv.2509.11399.
14. Yumou Fei, Dor Minzer, and Shuo Wang. Multi-pass streaming lower bounds for approximating max-cut. *arXiv preprint arXiv:2503.23404*, 2025. doi:10.48550/arXiv.2503.23404.
15. Oded Goldreich, Shari Goldwasser, and Dana Ron. Property testing and its connection to learning and approximation. *Journal of the ACM (JACM)*, 45(4):653–750, 1998. doi:10.1145/285055.285060.
16. Oded Goldreich and Dana Ron. A sublinear bipartiteness tester for bounded degree graphs. *Combinatorica*, 19(3):335–373, 1999. doi:10.1007/S004930050060.
17. Oded Goldreich and Dana Ron. Property testing in bounded degree graphs. *Algorithmica*, 32(2):302–343, 2002. doi:10.1007/S00453-001-0078-7.
18. Oded Goldreich and Dana Ron. On testing expansion in bounded-degree graphs. In *Studies in Complexity and Cryptography: Miscellanea on the Interplay between Randomness and Computation*, pages 68–75. Springer, 2011. doi:10.1007/978-3-642-22670-0_9.
19. Venkatesan Guruswami and Yuan Zhou. Tight bounds on the approximability of almost-satisfiable horn sat and exact hitting set. *Theory of Computing*, 8(1):239–267, 2012.
20. Johan Håstad. Some optimal inapproximability results. *Journal of the ACM (JACM)*, 48(4):798–859, 2001. doi:10.1145/502090.502098.

21 Michael Kapralov, Sanjeev Khanna, and Madhu Sudan. Streaming lower bounds for approximating max-cut. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1263–1282. SIAM, 2014.

22 Michael Kapralov and Dmitry Krachun. An optimal space lower bound for approximating max-cut. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, pages 277–288, 2019. doi:10.1145/3313276.3316364.

23 Subhash Khot. On the power of unique 2-prover 1-round games. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 767–775, 2002. doi:10.1145/509907.510017.

24 Prasad Raghavendra. Optimal algorithms and inapproximability results for every csp? In *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pages 245–254, 2008. doi:10.1145/1374376.1374414.

25 Raghuvansh R Saxena, Noah G Singer, Madhu Sudan, and Santhoshini Velusamy. Streaming algorithms via local algorithms for maximum directed cut. In *Proceedings of the 2025 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 3392–3408. SIAM, 2025. doi:10.1137/1.9781611978322.111.

26 Luca Trevisan. Non-approximability results for optimization problems on bounded degree instances. In *Proceedings of the thirty-third annual ACM symposium on Theory of computing*, pages 453–461, 2001. doi:10.1145/380752.380839.

27 Yuichi Yoshida. Optimal constant-time approximation algorithms and (unconditional) inapproximability results for every bounded-degree csp. In *Proceedings of the forty-third annual ACM symposium on Theory of computing*, pages 665–674, 2011. doi:10.1145/1993636.1993725.

28 Yuichi Yoshida and Hiro Ito. Query-number preserving reductions and linear lower bounds for testing. *IEICE transactions on information and systems*, 93(2):233–240, 2010. doi:10.1587/TRANSINF.E93.D.233.

29 Yuichi Yoshida and Hiro Ito. Property testing on k-vertex-connectivity of graphs. *Algorithmica*, 62(3):701–712, 2012. doi:10.1007/S00453-010-9477-Y.

A Definition of Bounded Width

In this section, we provide the definition and basic properties of the "bounded width" notion for CSPs. The class of bounded-width CSP templates is defined by the correctness of an algorithm, which we call the width-(k, ℓ) algorithm, in deciding satisfiability of instances.

▶ **Definition 14** ([12]). *Let k, ℓ be integers such that $\ell \geqslant k \geqslant 1$. A CSP template (D, Γ) is said to have width (k, ℓ) if for all instances $I \in \mathrm{CSP}(\Gamma)$, Algorithm 1 correctly decides whether I is satisfiable. A template (D, Γ) is said to have bounded width if it has width (k, ℓ) for some integers k, ℓ; otherwise, we say (D, Γ) has unbounded width.*

Note that Algorithm 1 is always correct if it outputs `unsatisfiable`, so the only mistake is when the input instance is unsatisfiable but it outputs `satisfiable`. Also, since the running time of Algorithm 1 is clearly polynomial in $|V|$ and m, we immediately have the following property:

▶ **Proposition 15.** *For a CSP template (D, Γ), we let Γ-$\textsc{Sat}$ be the problem of deciding satisfiability of instances from $\mathrm{CSP}(\Gamma)$. If (D, Γ) has bounded width, then $\Gamma\text{-}\textsc{Sat} \in \mathbf{P}$.*

Algorithm 1 Width-(k, ℓ) Algorithm.

Input : a CSP(Γ) instance $I = (V, (C_1, \ldots, C_m))$
Output : `satisfiable` or `unsatisfiable`

for *each subset* $U \subseteq V$ *of size* ℓ **do**
  Initialize a set of partial assignments $S_U \leftarrow D^U$
  for *each pair* (τ, i) *of partial assignment* $\tau \in D^U$ *and constraint index* $i \in [m]$ **do**
    if U *contains the scope of* C_i **and** τ *does not satisfy* C_i **then**
      $S_U \leftarrow S_U \setminus \{\tau\}$ `// remove inconsistencies with original constraints`

`// finish initialization; start main procedure`

repeat
  for *each pair of subsets* $U_1, U_2 \subseteq V$ *such that* $|U_1| = |U_2| = \ell$ **do**
    for *each (if any) subset* $W \subseteq U_1 \cap U_2$ *such that* $|W| = k$ **do**
      for *each* $\tau \in S_{U_1}$ **do**
        if $\tau_{|W} \neq \sigma_{|W}$ *for all* $\sigma \in S_{U_2}$ **then**
          $S_{U_1} \leftarrow S_{U_1} \setminus \{\tau\}$ `// remove inconsistencies in any` k `variables`

until *the sets* S_U *stop changing*
Output `unsatisfiable` if $S_U = \emptyset$ for some U; otherwise **output** `satisfiable`