Dichotomies for #CSP on Graphs That Forbid a Clique as a Minor

Boning Meng[1]
Institute of Software, University of Regensburg, Germany
University of Chinese Academy of Sciences, Beijing, China

Yicheng Pan[2]
State Key Laboratory of Complex & Critical Software Environment, Beihang University, Beijing, China

Abstract

We establish complexity dichotomies for #CSP and $\#R_D$-CSP problems (not necessarily symmetric) with Boolean domain and complex range over several key minor-closed graph classes. These dichotomies give a complete characterization of the complexity of #CSP and $\#R_D$-CSP over graph classes that forbid a complete graph as a minor. Notably, our algorithms apply not only to #CSP and $\#R_D$-CSP problems but also directly to the Holant setting.

2012 ACM Subject Classification Theory of computation → Problems, reductions and completeness

Keywords and phrases Graph Minor, Tree Decomposition, Computational Complexity, Counting, Constraint Satisfaction Problem, Holant, Matchgate

Digital Object Identifier 10.4230/LIPIcs.ESA.2026.47

Related Version *Full Version*: `https://arxiv.org/abs/2504.01354` [28]

Funding *Boning Meng*: Funded by the European Union (ERC, CountHom, 101077083). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council Executive Agency. Neither the European Union nor the granting authority can be held responsible for them.
Yicheng Pan: Supported by NSFC 61932002 and State Key Laboratory of Complex & Critical Software Environment (SKLCCSE-2024ZX-20).

1 Introduction

In this article, we study the counting constraint satisfaction problem (denoted as #CSP) with Boolean domain and complex-valued range over several typical minor-closed graph classes. #CSP is considered as one of the most typical frameworks in counting complexity, as it is capable of expressing a number of counting problems, such as counting vertex covers in a given graph (#VC) and counting solutions of a CNF formula (#SAT). The complexity of #CSP over general graphs has been fully classified on Boolean domain [17, 21, 2, 14] and general domain [3, 22, 1, 5, 4], and that over planar graphs has been fully classified on Boolean domain [26, 8, 29].

Restricted to the Boolean domain and complex-valued range, the only difference between the dichotomies for #CSP on general graphs and on planar graphs is the planar tractable case known as permutable-matchgate-transformable signatures, which can be generalized from counting perfect matchings in a natural way.

[1] First author.
[2] Corresponding author.

Counting perfect matchings (denoted as #PM) is the first natural counting problem proven to be #P-hard [36] on general graphs and polynomial-time computable on planar graphs (known as the FKT-algorithm) [27, 33]. The complexity of #PM has also been extensively studied on several minor-closed graph classes [38, 24, 34, 19, 32, 18, 23, 20, 35]. In particular, Thilikos and Wiederrecht presented a dichotomy for #PM on minor-closed graph classes [35] in 2022.

This motivates our study of dichotomies for #CSP (and its variant $\#R_D$-CSP($D \geq 3$), where each variable can only appear at most D times) on these minor-closed graph classes as well. In light of the aforementioned dichotomies, the key question is whether the results for #PM also hold for permutable matchgate signatures in the Holant setting.

This article provides a nuanced answer: if the arity of the signatures (which corresponds to the degree of vertices) is bounded, then the dichotomy for #PM from [35] still holds, thereby contributing to dichotomies for $\#R_D$-CSP($D \geq 3$). Otherwise, one can prove that the algorithm in [35] fails, and only simpler algorithms can be adapted, which contributes to dichotomies for #CSP. By combining these results, we establish dichotomies for both #CSP and $\#R_D$-CSP($D \geq 3$) on graph classes that exclude a clique as a minor (See Theorem 26 and 25 for details).

This paper is organized in order of dependency. In Section 2, we introduce the preliminaries needed in our proof. In Section 3, we present our main results. In Section 4, we decompose the main theorem into Lemma 28 – 31, which are sufficient conditions for it. We then prove Lemma 28 – 31 in Sections 5, 6, 8, 7, respectively.

Due to space limit, we omit the proof of lemmas labelled with (*). Please refer to the full version [28].

2 Preliminaries

2.1 #PM on minor-closed graph classes

We first establish key definitions needed for this study. An instance of #PM is a graph $G = (V, E)$ with weighted edges $w : E \to \mathbb{C}$. The weight of a matching M is $w(M) = \prod_{e \in M} w(e)$. The output of the instance is the sum of the weights of all the perfect matchings in G:

$$\#\mathrm{PM}(G) = \sum_{M : M \text{ is a perfect matching of } G} w(M)$$

When $w(e) = 1$ for each $e \in E$, the output of the instance is precisely the number of perfect matchings in the graph.

Contracting an edge (u, v) *in* G means replacing u, v with a new vertex w connected to all the neighbours of u, v. A graph H is a *minor* of G if H can be obtained from G by repeatedly deleting vertices, deleting edges, or contracting edges.

A graph class is a set of graphs, possibly infinite. A graph class $\mathcal{C}$ is *minor-closed* if it is closed under taking minors. If graph class $\mathcal{C}$ is defined by forbidding a finite set $\mathcal{G}$ of graphs as minors, then $\mathcal{G}$ is the *forbidden minor set* of $\mathcal{C}$, denoted by $\mathcal{C} = fb(\mathcal{G})$. The following theorem demonstrates how forbidden minor sets provide a powerful tool for characterizing graph classes.

▶ **Theorem 1** (Robertson-Seymour [30]). *If a graph class is minor-closed, then it has a finite forbidden minor set.*

Now, we introduce some notation. The graph class of all planar graphs is denoted by $\mathcal{PL}$. An apex graph is defined as a graph that can be embedded on the plane after the removal of a single vertex, which is referred to as the apex vertex of the graph. The graph class of all the apex graphs is denoted by $\mathcal{PLA}$.

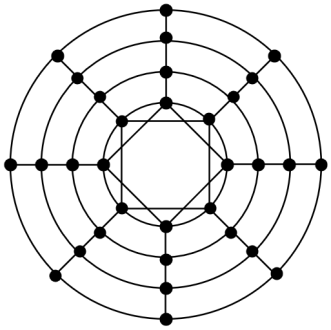


Figure 1 The shallow vortex grid of order 4, denoted as H_4.

We denote #PM on the graph class $\mathcal{C}$ as #PM$\langle\mathcal{C}\rangle$, and if $\mathcal{C} = fb(\mathcal{G})$, we denote #PM$\langle\mathcal{C}\rangle$ by #PM$[\mathcal{G}]$. Further, if $\mathcal{G} = \{G\}$, we denote the problem as #PM$[G]$ for convenience. [3] Using this notation, #PM$\langle\mathcal{PL}\rangle$, or equivalently #PM$[K_5, K_{3,3}]$, refer to #PM on planar graphs; #PM, or equivalently #PM$[\emptyset]$, refer to #PM on general graphs.

The dichotomy for #PM on minor-closed graphs can be stated as follows.

▶ **Definition 2** (Shallow vortex grid[35]). *The shallow vortex grid of order k is the graph that consists of:*

- *k cycles $C_1, \ldots, C_k$ of length $2k$, where $C_i = (v_{i,1}, v_{i,2}, \ldots, v_{i,2k}, v_{i,1}), 1 \leq i \leq k$;*
- *$2k$ paths $P_1, \ldots, P_{2k}$ of length k, where $P_j = (v_{1,j}, v_{2,j}, \ldots, v_{k,j}), 1 \leq j \leq 2k$;*
- *and 2 cycles C_{odd}, C_{even} of length k, where $C_{odd} = (v_{1,1}, v_{1,3}, \ldots, v_{1,2k-1}, v_{1,1})$ and $C_{even} = (v_{1,2}, v_{1,4}, \ldots, v_{1,2k}, v_{1,2})$.*

The graph class $\mathcal{H}$ is defined as $\{H \mid H$ is a minor of H_k for some $k \in \mathbb{N}^+\}$.

See Figure 1 for an example of shallow vortex grid.

▶ **Theorem 3** ([35]). *Let $\mathcal{G}$ be a finite set of graphs. If $\mathcal{G} \cap \mathcal{H} \neq \emptyset$, then #PM$[\mathcal{G}]$ can be solved in polynomial time; otherwise it is #P-hard.*

Central to Theorem 3 is the vga-hierarchy [35], a framework built upon a series of graph parameters associated with tree decompositions.

▶ **Definition 4** (Tree decomposition[31]). *A tree decomposition (T, β) of a graph $G = (V, E)$ consists of a tree T and a mapping $\beta : V(T) \to 2^{V(G)}$ that maps every node in T to a subset of $V(G)$ such that:*

- $\bigcup_{t \in T} \beta(t) = V(G)$
- $\forall uv \in E, \exists t \in T$ *such that* $u, v \in \beta(t)$
- $\forall v \in V$, $\{t \in T \mid v \in \beta(t)\}$ *induces a subtree in* T.

The *width* of a tree decomposition (T, β) is $\max_{t \in T}(|\beta(t)| - 1)$, and the *treewidth* of a graph $G = (V, E)$, denoted as $\mathsf{tw}(G)$, is the minimum width over all possible tree decompositions of G. For each $k \in \mathbb{N}^+$, the graph class of all graphs with treewidth no greater than k is defined to be $\mathcal{TW}_k = \{G \mid \mathsf{tw}(G) \leq k\}$. For $t \in T$, the *torso* of G at t is $G_t = (\beta(t), E_t)$, where $E_t = \{uv \in E \mid u, v \in \beta(t)\} \cup \{uv \mid u, v \in \beta(t), \exists dt \in T, u, v \in \beta(d)\}$. A tree decomposition is called a *normal tree decomposition* if for each $dt \in T$, neither $\beta(t) \subseteq \beta(d)$ nor $\beta(d) \subseteq \beta(t)$ holds.

[3] These notations also applies to other counting problems.

▶ **Lemma 5 (*).** *A normal tree decomposition can be obtained from an arbitrary tree composition (T, β) in $O(|T|^2|G|)$ time.*

▶ **Lemma 6 (*).** *For a normal tree decomposition (T, β) of G, $|T| \leq |G|$.*

A series of parameters can be defined based on tree decompositions. We confine ourselves to a self-contained fragment of definitions in [35] and make some modifications in order to provide a more intelligible explanation.

▶ **Definition 7.** *For a graph $G = (V, E)$ and a tree decomposition (T, β) of G, the ga3-width of (T, β) is the minimum k such that for every $t \in T$, if $|G_t| > k$, then G_t must satisfy the following conditions:*

- *There is an apex set $A_t \subseteq V(G_t)$, a surface Σ_t such that $G_t - A_t$ can be embedded on Σ_t without crossings;*
- *(g) Σ_t has Euler-genus at most k;*
- *(a) $|A_t| \leq k$;*
- *(3) For any $dt \in T$, $|\beta(d) \cap \beta(t) - A_t| \leq 3$. Further, if $|\beta(d) \cap \beta(t) - A_t| = 3$, then $\beta(d) \cap \beta(t) - A_t$ must bound a face in the embedding.*

We use $\mathsf{p}_{ga3}(G)$ to denote the minimum ga3-width over all possible tree decompositions of G.

Definition 7 can also define seven other parameters by independently replacing $g, a, 3$ with $-, -, +$. By replacing g and/or a with $-$, we set the k in the corresponding condition to 0. By replacing 3 with $+$, we delete the condition (3). To illustrate, $\mathsf{p}_{ga+} = k$ signifies the existence of a tree decomposition such that every torso of it can be embedded on a surface with Euler-genus at most k after deleting at most k vertices. In contrast, $\mathsf{p}_{--3} = k$ denotes the necessity for a tree decomposition whose torsos with size greater than k are planar and satisfy the (3) condition.

▶ Remark 8. The definitions presented here differ in two aspects from the origin vga-hierarchy in the article "Killing a vortex" [35]. Firstly, the concept of "vortex" is omitted, as it has already been "killed" and is not necessary in this article. Secondly, the (3) condition is introduced into the definitions, as it plays a pivotal role in the algorithms. For example, the parameter p_{ga+} in this article is equivalent to the parameter "p_{-ga}" in [35].

If a graph forbids certain minors, then it admits tree decompositions with certain properties, and these can be constructed in polynomial time.

▶ **Theorem 9** ([35]). *For a graph H which is a minor of H_k and a graph G, in $O(f(k) \cdot |G|^3)$ time where $f(k)$ is some computable function, we can find either the fact that H is a minor of G, or alternatively a tree decomposition (T, β) of G whose ga3-width is less than $c(k)$, where $c(k)$ is some computable function that depends only on k.*

A *single crossing graph* is defined as a graph that can be embedded on the plane with at most one crossing. Such graphs exhibit the following property:

▶ **Theorem 10** ([18]). *For a single crossing graph H, there exists a constant c such that for all graphs $G \in fb(\{H\})$, $\mathsf{p}_{--3}(G) \leq c$. Furthermore, a tree decomposition with $--3$-width less than c can be found in $O(|G|^4)$.*

2.2 Counting problems on planar graphs

Within various counting complexity frameworks, numerous #P-hard problems have been shown to become polynomial-time computable on planar graphs following the breakthrough of #PM [39, 15, 12, 13, 26, 8, 10, 9, 16, 7]. While most such cases are related to FKT-algorithm techniques, other algorithms are not subsumed by simply applying the FKT algorithm [10, 9, 7].

We now establish key definitions needed for this study. A *signature*, or a *constraint function*, is a function $f : \{0,1\}^k \to \mathbb{C}$, where k is the *arity* of f. For a string $\alpha = \alpha_1 \dots \alpha_k \in \{0,1\}^k$, the *Hamming weight* of α is the number of 1s in α. A signature f is said to be *symmetric* if the value of f depends only on the Hamming weight of the input string. A symmetric signature f of arity k can be denoted as $[f_0, f_1, ..., f_k]_k$, or simply $[f_0, f_1, ..., f_k]$ when k is clear from the context. Here, for $0 \le i \le k$, f_i is the value of f when the Hamming weight of the input string is i.

Two fundamental counting problem frameworks, #CSP and Holant, hold particular significance in counting complexity as they can express numerous important problems, e.g. counting vertex covers and #SAT. Furthermore, both #CSP and #PM can be formulated as Holant problems, indicating that Holant is a more generalized framework.

▶ **Definition 11.** *#CSP is defined by a signature (function) set $\mathcal{F}$, denoted as #CSP($\mathcal{F}$). An instance I of #CSP($\mathcal{F}$) has m variables and n signatures from $\mathcal{F}$ depending on these variables. The value of the instance then can be written as*

$$Z(I) = \sum_{(x_1,...,x_m)\in\{0,1\}^m} \prod_{1\le i\le n} f_i(x_{i_1}, ..., x_{i_k})$$

where $f_1, \dots, f_n$ are signatures in I and f_i depends on $x_{i_1}, ..., x_{i_k}$ for each $1 \le i \le n$.

The *underlying graph* of a #CSP($\mathcal{F}$) instance I is a bipartite (multi-)graph $G = (U, V, E)$, where for every constraint f there is a $u_f \in U$, for every variable x there is a $v_x \in V$, and $(u_f, v_x) \in E$ if and only if f depends on x. If we restrict the maximum degree of the vertices in V to be no more than a constant D, this kind of problem is denoted as $\#R_D$-CSP in [14].

▶ **Definition 12.** *Holant is defined by a signature (function) set $\mathcal{F}$, denoted as Holant($\mathcal{F}$). An instance of Holant($\mathcal{F}$) has an underlying (multi-)graph $G = (V, E), |V| = n, |E| = m$. Each vertex $v \in V$ is assigned a signature from $\mathcal{F}$ and each edge in E represents a variable. The signature assigned to the vertex v is denoted as f_v. The value of the instance then can be written as*

$$Z(G) = \sum_{(e_1,...,e_m)\in\{0,1\}^m} \prod_{v\in V} f_v(e_{v_1}, ..., e_{v_k})$$

where $e_{v_1}, ..., e_{v_k}$ are incident to v.

Furthermore, we use Holant($\mathcal{F}_1 \mid \mathcal{F}_2$) to represent Holant($\mathcal{F}_1 \cup \mathcal{F}_2$) with the restriction that the underlying graph $G = (U, V, E)$ is bipartite, and each vertex $u \in U$ is assigned a signature from $\mathcal{F}_1$ while each vertex $v \in V$ is assigned a signature from $\mathcal{F}_2$. We denote by $\mathcal{EQ}$ the set of all equality functions. In other words, $\mathcal{EQ} = \{=_k \mid k \ge 1\}$ where $=_k$ is the signature $[1, 0, ..., 0, 1]_k$. We also denote $\{=_k \mid 1 \le k \le D\}$ by $\mathcal{EQ}_{\le D}$ for each integer $D \ge 3$. We use $\le_T$ and $\equiv_T$ to denote polynomial-time Turing reduction and equivalence respectively. By definition, we have the following lemma.

Table 1 References related to #PM and dichotomies for #CSP, $\#R_D$-CSP and Holant problems over sorts of graphs. Our results fill the blank of #CSP and $\#R_D$-CSP over minor-free graphs.

	General graphs	Planar graphs	Minor free graphs
#PM	[36]	[27, 33]	[35, 20]
sym-#CSP	[14]	[26]	**Our results**
#CSP		[8]	
$\#R_D$-CSP($D \geq 3$)		[29]	
sym-Holant	[11]	[9]	open
Holant	open	open	open

▶ **Lemma 13.** *Let $\mathcal{C}$ be an arbitrary graph class, $\mathcal{F}$ be an arbitrary signature set and $D \geq 3$ be an integer. Then,*

$$\#CSP(\mathcal{F})\langle\mathcal{C}\rangle \equiv_T Holant(\mathcal{F} \mid \mathcal{EQ})\langle\mathcal{C}\rangle, \quad \#R_D\text{-}CSP(\mathcal{F})\langle\mathcal{C}\rangle \equiv_T Holant(\mathcal{F} \mid \mathcal{EQ}_{\leq D})\langle\mathcal{C}\rangle.$$

Several dichotomies have been demonstrated on general graphs and planar graphs in the study of #CSP and Holant problems, stating that for each problem parameterized by $\mathcal{F}$, either it is polynomial-time computable or it is #P-hard. Please refer to Table 1 for corresponding references and the position of our results.

We first introduce relevant tractable cases. $\mathscr{A}$ and $\mathscr{P}$ are 2 well-known tractable cases for #CSP on general graphs [14].

▶ **Definition 14.** *A signature f with arity k is of affine type if it has the form:*

$$\lambda\chi_{AX=b} \cdot \mathrm{i}^{\sum_{1\leq i\leq n} a_i {x_i}^2 + 2\sum_{1\leq i<j\leq n} b_{ij}x_ix_j}$$

where i denotes the imaginary unit, $a_i, b_{ij} \in \{0,1\}, \lambda \in \mathbb{C}$, $AX = b$ is a system of linear equations on $\mathbb{Z}_2$ and

$$\chi_{AX=b} = \begin{cases} 1, & \text{if } AX = b; \\ 0, & \text{otherwise.} \end{cases}$$

$\mathscr{A}$ denotes the set of all the signatures of affine type.

▶ **Definition 15.** *A signature f is an $\mathcal{E}$-signature if it has value 0 except for 2 complementary supports. A signature f with arity k is of product type if it can be expressed as a tensor product of $\mathcal{E}$-signatures.*

$\mathscr{P}$ denotes the set of all the signatures of product type.

$\widehat{\mathscr{M}_P}$ is the set $\mathscr{M}_P$ of permutable signatures under a specific holographic transformation. The set $\mathscr{M}_P$ is defined as follows:

▶ **Definition 16.** *A matchgate is a planar graph $G = (V, E)$ with weighted edges $w : E \to \mathbb{C}$, and together with some external nodes $U \subseteq V$ on its outer face labeled by $\{1, 2, ..., |U|\}$ in a clockwise order. The signature f of a matchgate G is a Boolean signature of arity $|U|$ and for each $\alpha \in \{0,1\}^{|U|}$,*

$$f(\alpha) = \#PM(G - X),$$

where $X \subseteq U$ and a vertex in U with label i belongs to X if and only if the ith bit of α is 1.

A signature f is a matchgate signature if it is the signature of some matchgate. $\mathscr{M}$ denotes all the matchgate signatures.

Notably, variables of a matchgate signature can only be placed in a specific order, and to remove this restriction, permutable matchgate signatures are defined [29].

▶ **Definition 17.** *Suppose f is a signature of arity n. For a permutation $\pi \in S_n$, we use f_π to denote the signature*

$$f_\pi(x_1, \ldots, x_n) = f(\pi(x_1), \ldots, \pi(x_n))$$

If for each $\pi \in S_n$, f_π is a matchgate signature, we say f is a permutable matchgate signature.

We use $\mathscr{M}_P$ to denote the set of all the permutable matchgate signatures.

It is noteworthy that Holant$\langle \mathcal{PL} \rangle$ problems defined by permutable matchgate signatures are tractable. Indeed, by replacing each signature with its corresponding matchgate, the problem can be transformed into an instance of #PM$\langle \mathcal{PL} \rangle$, which has the same output and can be computed efficiently using the FKT algorithm. In this sense, permutable matchgate signatures can be viewed as a natural generalization that captures and extends the local constraints inherent in #PM.

The symbol $\widehat{\ }$ denotes a specific holographic transformation defined by $H_2 = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$. The following theorem demonstrates the relationship between the initial and transformed problems:

▶ **Theorem 18** (Holographic Transformation[37, 6]). *$Holant(\mathcal{F} \mid \mathcal{G}) \equiv_T Holant(\mathcal{F}T^{-1} \mid T\mathcal{G})$*

As $H_2^{-1} = \frac{1}{2}H_2$, by Theorem 18 we have: $Holant(\mathcal{F} \mid \mathcal{G}) \equiv_T Holant(\widehat{\mathcal{F}} \mid \widehat{\mathcal{G}})$. It is noteworthy that equality signatures can be transformed into permutable matchgate signatures under this transformation.

▶ **Lemma 19.** *For each $k \geq 1$, $\widehat{=_k} = [1, 0, 1, 0, 1, 0, \ldots]_k \in \mathscr{M}_P$. Consequently, $\widehat{\mathcal{EQ}} = \{[1, 0, 1, 0, 1, 0, \ldots]_k \mid k \geq 1\}$ and for $D \geq 3$, $\widehat{\mathcal{EQ}_{\leq D}} = \{[1, 0, 1, 0, 1, 0, \ldots]_k \mid 1 \leq k \leq D\}$.*

We now present the most relevant dichotomies and explain the explicit tractable case:

▶ **Theorem 20** ([14]). *If $\mathcal{F} \subseteq \mathscr{A}$ or $\mathcal{F} \subseteq \mathscr{P}$, #CSP($\mathcal{F}$) is computable in polynomial time; otherwise it is #P-hard.*

▶ **Theorem 21** ([14]). *Suppose $D \geq 3$ is an integer. If $\mathcal{F} \subseteq \mathscr{A}$ or $\mathcal{F} \subseteq \mathscr{P}$, #$R_D$-CSP($\mathcal{F}$) is computable in polynomial time; otherwise it is #P-hard.*

▶ **Theorem 22** ([8, 29]). *If $\mathcal{F} \subseteq \mathscr{A}$ or $\mathcal{F} \subseteq \mathscr{P}$ or $\mathcal{F} \subseteq \widehat{\mathscr{M}_P}$, #CSP($\mathcal{F}$)$\langle \mathcal{PL} \rangle$ is computable in polynomial time; otherwise it is #P-hard.*

▶ **Theorem 23** ([29]). *Suppose $D \geq 3$ is an integer. If $\mathcal{F} \subseteq \mathscr{A}$ or $\mathcal{F} \subseteq \mathscr{P}$ or $\mathcal{F} \subseteq \widehat{\mathscr{M}_P}$, #$R_D$-CSP($\mathcal{F}$)$\langle \mathcal{PL} \rangle$ is computable in polynomial time; otherwise it is #P-hard.*

▶ **Theorem 24** ([25]). *For each $k \in \mathbb{N}^+$, if $\mathcal{F}$ is finite, #CSP($\mathcal{F}$)$\langle \mathcal{TW}_k \rangle$ is computable in $c^{O(k)}|I|$ time, where c is only parameterized by $\mathcal{F}$.*

3 Our results

Since #PM can be expressed as a Holant problem, it is natural to investigate the complexity of Holant problems on minor-closed graph classes. A standard approach to studying the complexity of Holant problems begins with the framework of #CSP, which typically assumes that all equality signatures are available.

Our work resolves the complexity of #CSP for key minor-closed graph classes previously identified as fundamental cases in the study of #PM [18, 20, 35]. We establish complete complexity dichotomies for these graph classes. This represents the first counting dichotomy simultaneously incorporating both signature sets and graph classes, unifying their separate complexity classifications.

▶ **Theorem 25.** *Let $\mathcal{F}$ be a finite signature set on Boolean domain with complex range and n be the number of vertices in the underlying graph. Then:*

1. *For each $k \in \mathbb{N}^+$, $\#CSP(\mathcal{F})\langle \mathcal{TW}_k \rangle$ can be computed in $c^{O(k)}n$ time, where c is only parameterized by $\mathcal{F}$.*
2. *Suppose that H is a single crossing graph. $\#CSP(\mathcal{F})[H]$ is computable in polynomial time if $\mathcal{F} \subseteq \mathscr{A}$ or $\mathcal{F} \subseteq \mathscr{P}$ or $\mathcal{F} \subseteq \widehat{\mathscr{M}_P}$; otherwise it is #P-hard.*
3. *$\#CSP(\mathcal{F})\langle \mathcal{PLA} \rangle$ is computable in polynomial time if $\mathcal{F} \subseteq \mathscr{A}$ or $\mathcal{F} \subseteq \mathscr{P}$; otherwise it is #P-hard (recall that $\mathcal{PLA}$ denotes apex graphs).*
4. *Suppose that $\mathcal{G} \cap \mathcal{H} \neq \emptyset$, $\mathcal{G} \cap \mathcal{PL} = \emptyset$. Then for any integer $D \geq 3$, $\#R_D\text{-}CSP(\mathcal{F})[\mathcal{G}]$ is computable in polynomial time if $\mathcal{F} \subseteq \mathscr{A}$ or $\mathcal{F} \subseteq \mathscr{P}$; it is computable in $f(k)\mathrm{poly(n)}$ time for some computable f, where k is only parameterized by D and $\mathcal{F}$, if $\mathcal{F} \subseteq \widehat{\mathscr{M}_P}$; otherwise, it is #P-hard.*
5. *Suppose that $\mathcal{G} \cap \mathcal{H} = \emptyset$. Then for any integer $D \geq 3$, $\#R_D\text{-}CSP(\mathcal{F})[\mathcal{G}]$ is computable in polynomial time if $\mathcal{F} \subseteq \mathscr{A}$ or $\mathcal{F} \subseteq \mathscr{P}$; otherwise, it is #P-hard.*

Informally speaking, the graph class in the second statement corresponds to the case where every local part of the tree decomposition can be embedded in a plane. The third statement corresponds to the presence of an apex vertex, while the fourth describes cases where, after removing a constant number of apex vertices, every local part of the tree decomposition can be embedded on a surface of bounded genus. The fifth statement corresponds to the existence of a vortex. It can be observed that vortices and apex vertices serve as two critical boundaries in the complexity classification.

All of the graph classes listed in Theorem 25 are minor-closed classes by Theorem 1, though it might be challenging to specify the exact forbidden minor sets for some of them. Theorem 25 can also be applied to #CSP problems that forbid a complete graph as a minor.

▶ **Theorem 26.** *Let $\mathcal{F}$ be a finite signature set on Boolean domain with complex range. Then:*

1. *$\#CSP(\mathcal{F})[K_4]$ can be computed in linear time;*
2. *$\#CSP(\mathcal{F})[K_5]$ is computable in polynomial time if $\mathcal{F} \subseteq \mathscr{A}$ or $\mathcal{F} \subseteq \mathscr{P}$ or $\mathcal{F} \subseteq \widehat{\mathscr{M}_P}$; otherwise it is #P-hard.*
3. *$\#CSP(\mathcal{F})[K_6]$ is computable in polynomial time if $\mathcal{F} \subseteq \mathscr{A}$ or $\mathcal{F} \subseteq \mathscr{P}$; otherwise it is #P-hard.*
4. *For any integer $D \geq 3$, $\#R_D\text{-}CSP(\mathcal{F})[K_7]$ is computable in polynomial time if $\mathcal{F} \subseteq \mathscr{A}$ or $\mathcal{F} \subseteq \mathscr{P}$; it is computable in $f(k)\mathrm{poly(n)}$ time for some computable f, where k is only parameterized by D and $\mathcal{F}$, if $\mathcal{F} \subseteq \widehat{\mathscr{M}_P}$; otherwise, it is #P-hard.*
5. *For any integer $D \geq 3$, $\#R_D\text{-}CSP(\mathcal{F})[K_8]$is computable in polynomial time if $\mathcal{F} \subseteq \mathscr{A}$ or $\mathcal{F} \subseteq \mathscr{P}$; otherwise it is #P-hard.*

Theorem 26's results are summarized in Table 2. We emphasize that Theorem 26 represents a simplified version of our complete results in Theorem 25, with each statement being directly derivable as a corollary due to the following facts.

Table 2 A summary of Theorem 26. Each row denotes a certain case for $\mathcal{F}$. Each column denotes the forbidden minors of the graph class that the underlying graph is restricted to. The symbol "P" denotes #CSP and $\#R_D$-CSP$(D \geq 3)$ are polynomial-time computable, while "#P" denotes #CSP and $\#R_D$-CSP$(D \geq 3)$ are #P-hard. In particular, "#P,P" denotes #CSP is #P-hard, but $\#R_D$-CSP$(D \geq 3)$ are polynomial-time computable. We use (1)-(5) to indicate the conclusions corresponding to the five statements in Theorem 26.

Forbidden minors	K_4	$K_5, K_{3,3}$	K_5	K_6	K_7	K_8
$\mathcal{F} \subseteq \mathscr{A}$	P [14]					
$\mathcal{F} \subseteq \mathscr{P}$						
$\mathcal{F} \subseteq \widehat{\mathscr{M}_P}$, and $\mathcal{F} \nsubseteq \mathscr{A}, \mathscr{P}$	P [25](1)	P [8, 29]	P (2)	#P,P (3),(4)	#P (5)	
Otherwise		#P [8, 29]				

▶ **Lemma 27.** *The following statements hold:*

1. $fb(\{K_4\}) = \mathcal{TW}_2$;
2. K_5 *is a single crossing graph;*
3. $\mathcal{PLA} \subseteq fb(\{K_6\})$;
4. K_7 *is a minor of* H_{18} *and can not be embedded on the plane [35];*
5. $K_8 \notin \mathcal{H}$ *[35].*

It is a bit surprising that an unbounded maximum degree of vertices will turn over the complexity of #CSP problem on certain graph classes. This phenomenon has never been observed in the previous study. Here, we explain this phenomenon informally. When the maximum degree is not bounded, we may not enumerate all the valid assignments of edges incident to a single vertex, as the number of these assignments can be exponential in $|G|$. Consequently, the deletion of a single vertex from G, as performed in the algorithm described in [35], is no longer an available approach, since it would result in an exponential increase in the time complexity. Actually, by the third statement in Theorem 25, we prove that even the introduction of a single apex vertex would induce #P-hardness. This means that for #CSP problem with unbounded degree, after "killing a vortex", we must further "kill an apex" (Definition 7), to obtain a polynomial-time algorithm.

We emphasize that Theorems 25 remain valid for non-symmetric signatures in $\mathcal{F}$. This extension builds on the comprehensive characterization of permutable matchgate signatures in [29]. Our proof leverages these results to convert general #CSP cases to symmetric cases (sym-#CSP) in the algorithm part and reduce sym-#CSP problems back to general #CSP in the hardness proof.

Furthermore, in the proof of Statement 2 (respectively, Statement 4), we develop an algorithm that is not restricted to the #CSP setting (Lemma 28 and 29). It also works for Holant problems defined by permutable matchgate signatures of unbounded (resp., bounded) arity. This constitutes a non-trivial generalization of the prior algorithms designed for #PM.

4 Preprocessing

The Statement 1 in Theorems 25 (corresponding to the "K_4" column in Table 2) follows directly from Theorem 24 [25].

For Statements 2-5 in Theorem 25, we address the following analysis for #CSP, and the same arguments hold for $\#R_D$-CSP$(D \geq 3)$ as well by replacing $\mathcal{EQ}$ with $\mathcal{EQ}_{\leq D}$.

- By Theorem 20, if $\mathcal{F} \subseteq \mathscr{A}$ or $\mathcal{F} \subseteq \mathscr{P}$, then #CSP$(\mathcal{F})\langle\mathcal{C}\rangle$ is computable in polynomial time for arbitrary $\mathcal{C}$.

- By Theorem 22, if none of $\mathcal{F} \subseteq \mathscr{A}$, $\mathcal{F} \subseteq \mathscr{P}$ and $\mathcal{F} \subseteq \widehat{\mathscr{M}_P}$ holds and $\mathcal{PL} \subseteq \mathcal{C}$, then $\#\mathrm{CSP}(\mathcal{F})\langle\mathcal{C}\rangle$ is #P-hard.

Moreover, it can be verified that $(\widehat{\mathscr{M}} - \mathscr{A}) \cap \mathscr{P} = \emptyset$ [8]. Consequently, for a graph class $\mathcal{C}$ satisfying $\mathcal{PL} \subseteq \mathcal{C}$, the only unknown case is when $\mathcal{F} \subseteq \widehat{\mathscr{M}_P}$, but $\mathcal{F} \nsubseteq \mathscr{A}$.

By Lemma 13 and Theorem 18, $\#\mathrm{CSP}(\mathcal{F})\langle\mathcal{C}\rangle \equiv_T \mathrm{Holant}(\mathcal{F} \mid \mathcal{EQ})\langle\mathcal{C}\rangle \equiv_T \mathrm{Holant}(\widehat{\mathcal{F}} \mid \widehat{\mathcal{EQ}})\langle\mathcal{C}\rangle$. As $\widehat{\mathcal{EQ}} = \{[1,0,1,0,...]_k \mid k \geq 1\}$ by Lemma 19, both $\widehat{\mathcal{F}}, \widehat{\mathcal{EQ}} \subseteq \mathscr{M}_P$. Consequently, to show the unknown case is polynomial-time computable (Statement 2 and 4), it is sufficient to give an algorithm for Holant problems defined by permutable matchgate signatures:

▶ **Lemma 28.** *If $\mathcal{F} \subseteq \mathscr{M}_P$ (with unbounded maximum degree), then for any single crossing graph H, there is an algorithm that computes Holant$(\mathcal{F})[H]$ in polynomial time.*

▶ **Lemma 29.** *Suppose that $\mathcal{F} \subseteq \mathscr{M}_P$ and all the signatures in $\mathcal{F}$ are of arity at most k. Then for any graph class $\mathcal{G}$ satisfying $\mathcal{G} \cap \mathcal{H} \neq \emptyset$, there is an algorithm that computes Holant$(\mathcal{F})[\mathcal{G}]$ in polynomial time.*

Besides, as $\widehat{\mathscr{A}} = \mathscr{A}$, there exists a signature $f \in \widehat{\mathcal{F}}$ satisfying $f \in \mathscr{M}_P - \mathscr{A}$. Consequently, to show the unknown case is #P-hard (Statement 3 and 5), it is sufficient to prove #P-hardness based on f in the bipartite Holant setting.

▶ **Lemma 30.** *If $f \in \mathscr{M}_P - \mathscr{A}$, then Holant$(\{f\} \mid \widehat{\mathcal{EQ}})\langle\mathcal{PLA}\rangle$ is #P-hard.*

▶ **Lemma 31.** *If $\mathcal{G} \cap \mathcal{H} = \emptyset$, and $f \in \mathscr{M}_P - \mathscr{A}$, then Holant$(\{f\} \mid \widehat{\mathcal{EQ}_{\leq 3}})[\mathcal{G}]$ is #P-hard.*

In Sections 5 – 8, we prove Lemma 28, 29, 31, 30 respectively.

5 An algorithm for #CSP on graphs that forbids a single crossing minor

In this section, we prove Lemma 28.

In order to provide a more comprehensive understanding of the algorithm, we will now introduce a number of additional definitions. Pick a node $r \in T$ as the root of the tree. For two nodes $s, t \in T$, $s \leq t$ means that t is an ancestor of s. For a node $t \in T$, let $G_{\leq t}$ be the graph induced by $\bigcup_{s \leq t} \beta(s)$. For an edge $dt \in T$, if t is a child of d, we denote $\beta(t) \cap \beta(d)$ as the *navel* of $G_{\leq t}$, denoted as X_t. The navel of $G_{\leq r}$ is defined as $\emptyset$. The algorithm runs as follows:

1. Find a normal tree decomposition (T', β') with $g-3$-width no greater than h, and arbitrarily decide an $r' \in T'$ as the root;
2. Modify (T', β') to another tree decomposition (T, β) with a root r and of $g-3$-width no greater than h, such that for each $t \in T$, the children of t do not have the same navel.
3. The effect of a leaf is equivalent to some matchgate signature defined by a so-called representative function, so the final output can be obtained by computing the representative functions and replacing the leaves with corresponding matchgate signatures recursively.

For Step 1, by Theorem 10, there exist a constant h such that a tree decomposition (T', β') of G with $g-3$-width less than h can be computed in $O(|G|^4)$ time for arbitrary G. We may further assume the tree decomposition is normal as the size of the tree decomposition is $O(|G|^4)$ and we can obtain a normal tree decomposition in polynomial time by Lemma 5.

We noted that we actually compute a tree decomposition (T', β') with $--3$-width less than h in Step 1. In contrast, Step 2,3 in the algorithm only need a tree decomposition of bounded $g-3$-width, which means that the algorithm has the potential to solve a wider range of problems.

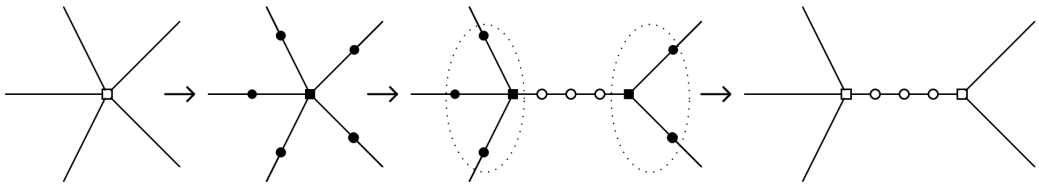


Figure 2 A visualization of the construction of a (S_1, S_2)-path gadget of f. Hollow squares: permutable matchgate signatures; Solid squares: symmetric matchgate signatures; Solid circles: edge signatures; Hollow circles: binary signatures in the path gadget. The construction has 3 steps: 1. realize f with a symmetric g and edge signatures, 2. replace g with a path gadget, 3. contract the edge signatures to the ends of the path.

Step 2 can be done in polynomial time (*). For Step 3, the key observation is that each permutable matchgate signature can be realized by a path gadget.

▶ **Lemma 32 (*).** *Suppose $f \in \mathscr{M}_P$ and (S_1, S_2) is a partition of its variables. Then there exists a path gadget PP (as shown in the rightmost panel of Figure 2) consisting of permutable matchgate signatures that realizes f, with dangling edges representing variables in S_1 and S_2 connecting to the two distinct ends of the gadget, respectively.*

The proof of Lemma 32 uses the characterization of permutable matchgate signatures from [29]. A visualization of the proof sketch is also provided in Figure 2.

Now we present how the algorithm works on a leaf node $t \in T$ whose parent is $d \in T$ in the tree decomposition, as shown in Figure 3. Figure 3a shows the graph induced by $\beta(t) \cup \beta(d)$. By replacing each vertex x in the navel X_t with a corresponding path gadget (Figure 3b) with 2 ends x_d and x_t, the graph induced by the vertices in $(\beta(t) - \beta(d)) \cup \{x_t \mid x \in X_t\}$ forms a gadget with at most three dangling edges (Figure 3c). We denote the signature realized by this gadget as the representative signature of $G_{\leq t}$. The representative signature can be computed efficiently due to the following lemma:

▶ **Lemma 33 (*).** *Suppose G is a graph which can be embedded on a surface with genus at most h, and each vertex of G is assigned a permutable matchgate signature. Then $Z(G)$ can be computed in polynomial time.*

Consequently, we may replace the gadget with a single vertex assigned the representative signature, as shown in Figure 3d,3e.

Now we show this procedure can be done recursively. We first show that the representative signature is a permutable matchgate signature. This is because the signature is realized via permutable matchgate signatures, it satisfies the parity condition: it is non-zero only on inputs with even (or odd) Hamming weight. By the construction in [37, Proposition 6.1 and 6.2], we have the following lemma and prove that the representative signature is a permutable matchgate signature.

▶ **Lemma 34.** *If a signature f of arity $k \leq 3$ satisfy the parity condition, then f is a permutable matchgate signature.*

Recall that the $g-3$-width of (T, β) is h. Due to the construction in Step 2 and property (3) of the tree decomposition, it can be shown that the maximum $g-3$-width of the resulting graphs encountered throughout the algorithm is at most $O(h^3)$, which can be regarded as a constant (*). Consequently, this procedure can be carried out recursively and finally outputs $Z(G)$.

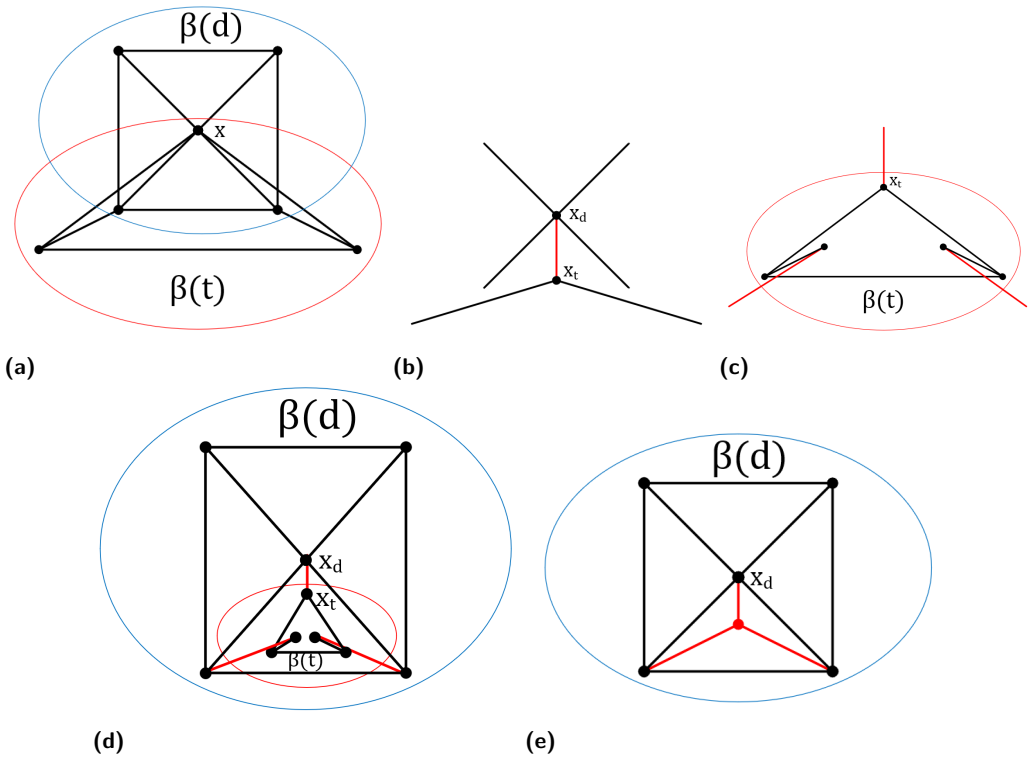


Figure 3 An example of the procedure of the algorithm. (a) The graph induced by $\beta(t) \cup \beta(d)$ where t is a leaf and a child of d. $\beta(t)$ is enclosed by the red ellipse while $\beta(d)$ is enclosed by the blue ellipse. (b) The path gadget for x, where the path is colored in red and 3 vertices of degree 2 on the path are omitted. Other vertices in the navel can be replaced similarly. (c) The construction of the gadget $GG_{\leq t}$, where the red edges represent the dangling edges in $D_{\leq t}$. (d)(e) Visualization of how $G_{\leq t}$ is replaced by a single vertex in $G_{\leq d}$. The introduced vertex is colored in red as well as the remaining parts of the path gadgets.

6 An algorithm for bounded degree #CSP on minor-free graphs

In this section, we prove Lemma 29. The algorithm proceeds as follows:

1. Find a tree decomposition (T, β) with $ga3$-width $\leq h$, decide an $r \in T$ as the root;
2. Compute boundary mappings from leaves to the root.

The algorithm is analogous to that described in Section 5. The algorithm was developed on the basis of the following principle: given that the degree of each vertex in G is bounded, it is now possible to deal with graphs with apex vertices, which enables the development of a more generalized algorithm.

We now focus on Step 1. For a graph $H \in \mathcal{G} \cap \mathcal{H}$, let $h(H)$ be the minimum integer t such that H is a minor of H_t. Let $h' = \min_{H \in \mathcal{G} \cap \mathcal{H}} h(H)$, which is computable. For each graph $G \in fb(\mathcal{G})$, G does not contain a $H_{h'}$ minor, otherwise a contradiction would ensue. Then a tree decomposition (T, β) of G with $ga3$-width less than h can also be computed in polynomial time by Theorem 9. We may further assume the tree decomposition is normal by Lemma 5.

Again, we introduce some more definitions to illustrate Step 2. For an edge $dt \in T$, if t is a child of d, we call $\beta(t) \cap \beta(d) - A_d$ the *navel* of $G_{\leq t}$, denoted as X_t. [4] The *boundary edge set* B_t of $G_{\leq t}$ is all the edges in $G_{\leq t}$ with one endpoint in $\beta(t) \cap \beta(d)$ and the other one in $V(G_{\leq t}) - \beta(t) \cap \beta(d)$. In particular, both X_r and B_r are defined as $\emptyset$. We also use Q_t to denote the set of all the edges incident to some $a \in A_t$ and use P_t to denote $Q_t - B_t$. As the maximum degree is bounded by k and the size of $X_t \cup A_t$ is bounded by $h + 3$, $|B_t \cup P_t| \leq k(h + 3)$.

For $dt \in T$ and t is a child of d, we want to record the value of $G_{\leq t}$ when the assignment of edges in B_t is given. This would allow us to care only about the value of edges in B_t instead of the inner structure of $G_{\leq t}$.

▶ **Definition 35.** *For $dt \in T$ and t is a leaf, the boundary gadget of $G_{\leq t}$ is denoted as $GG_{\leq t}$ which is the gadget induced by $V(G_{\leq t}) - \beta(t) \cap \beta(d)$. The boundary mapping of $G_{\leq t}$ is a mapping $BM_t : \{0,1\}^{|B_t|} \to \mathbb{C}$ such that for each $\sigma \in \{0,1\}^{|B_t|}$, $BM_t(\sigma) = Z(GG^{\sigma}_{\leq t})$.*

The computation of the boundary mapping is based on the following lemma:

▶ **Lemma 36 (*).** *Suppose G is a graph with maximum degree k, and there exists $A \subseteq V(G)$ with $|A| \leq h$, such that $G - A$ can be embedded on a surface with genus at most h, and each vertex of G is assigned a permutable matchgate signature, then the value of G can be computed in $f(k)\mathrm{poly(n)}$ time for some computable f.*

Then we are able to compute the boundary mapping of a leaf using Lemma 36:

▶ **Lemma 37 (*).** *Suppose $t \in T$ is a leaf, then the boundary mapping of $G_{\leq t}$ can be computed in $f(k)\mathrm{poly(n)}$ time for some computable f.*

Just like the representative signatures, boundary mappings can also be computed recursively and efficiently (*). See Figure 4 for a visualization.

7 Hardness for $\#R_D$-CSP on minor-free graphs

In this section, we prove Lemma 31. We commence with the definition of the graph class $\mathcal{R}$:

▶ **Definition 38** ([35]). *A ring-blowup of a planar graph $G = (V, E)$ is a graph $G' = (V', E')$ obtained from a planar embedding of G by adding true twins for each vertex on the outer face. That is, we denote the set of vertices on the outer face in the embedding as $V_f \subseteq V$ and let $V' = V \cup V'_f$, where $V'_f = \{u_v | v \in V_f\}$ is a copy of V_f. For edges, vertices in V'_f are incident to its original, its original's neighbor, and its original's neighbor's copies, i.e., $E' = E \cup \{(u_v, w) | (v, w) \in E\} \cup \{(u_v, u_w) | (v, w) \in E\} \cup \{(u_v, v) | v \in V_f\}$.*

A graph G is called a ring-blowup graph if it is a subgraph of a ring-blowup of some planar graph. We use $\mathcal{R}$ to denote the class of ring-blowup graphs.

An outcrossing graph is a graph that only has crossings on its outer face. Every outcrossing graph belongs to $\mathcal{R}$. See Figure 5a for an example. Our starting point is the fact that Holant$([0, 0, 1, 0])$ is #P-hard [40].

By slightly modifying the method in [20], we can prove #P-hardness on outcrossing graph and achieve the following lemma:

▶ **Lemma 39 (*).** *Holant$([0, 0, 1, 0]) \leq_T$ Holant$([0, 0, 1, 0])\langle\mathcal{R}\rangle$*

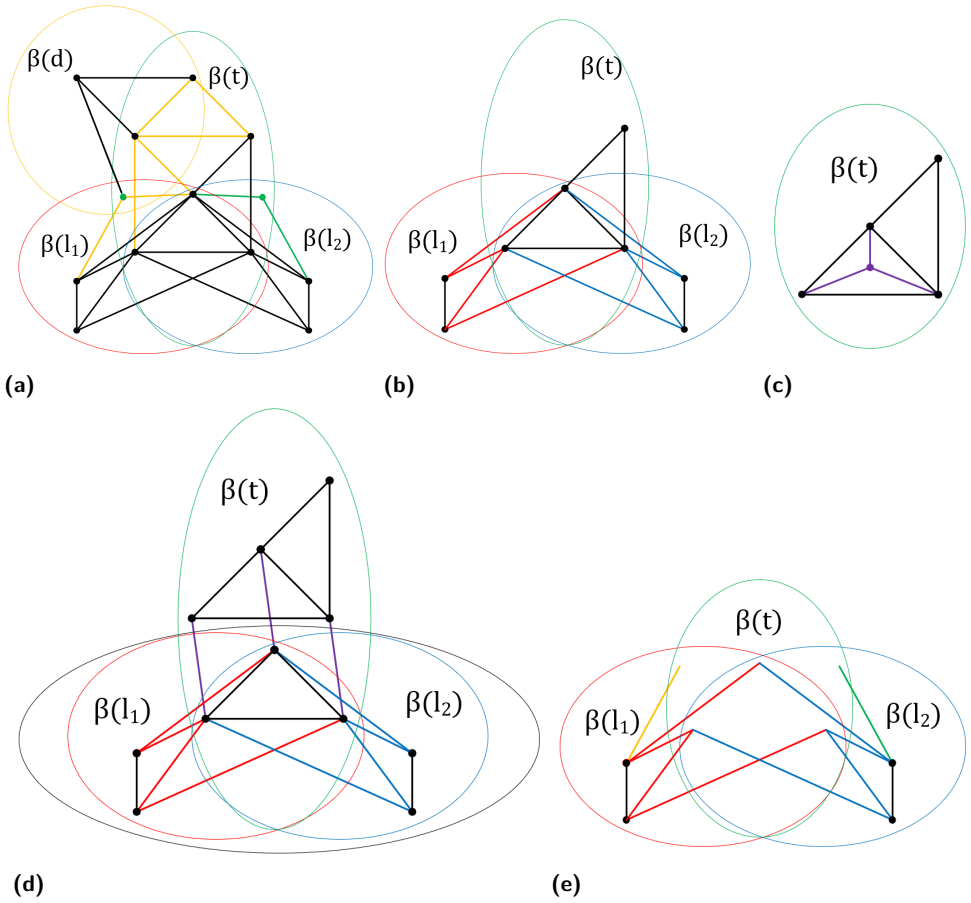


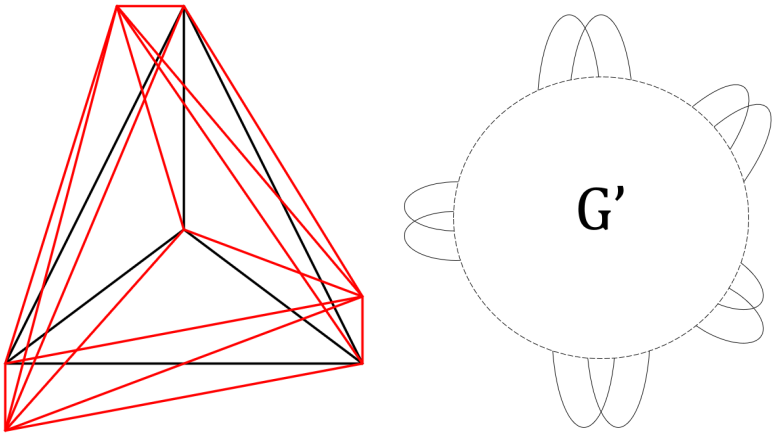
Figure 4 An example of the procedure of the algorithm. (a) The graph induced by $\beta(d) \cup \beta(t) \cup \beta(l_1) \cup \beta(l_2)$. $\beta(d), \beta(t), \beta(l_1), \beta(l_2)$ are enclosed by the respective orange, green, red, and blue ellipses. Edges in B_t are coloured in orange, while vertices in A_t and edges in P_t are coloured in green. l_1, l_2 have the same navel in $X \subseteq \beta(t)$(b) The figure of $GG^{\sigma}_{\leq t}$ obtained by removing apex vertices and navel vertices in $\beta(t)$. Edges belonging to B_{l_1} are coloured in red while those belonging to B_{l_2} are coloured in blue.(c) The figure of $HH^{\sigma}_{\leq t}$ obtained by replacing $\beta(l_1) \cup \beta(l_2) - X$ with a single vertex. The introduced v_l is coloured in violet. (d) The figure of $GH^{\sigma}_{\leq t}$ obtained by replacing vertices in X with path gadgets. The gadget $LL^{\sigma}_{\leq l}$ is enclosed by the black ellipse and the path gadgets are coloured in violet. (e) Visualization of the fact that $B_{l_1} \cup B_{l_2} \subseteq B_t \cup P_t \cup E_{X_l}$.

As proved in [35], $\mathcal{R}$ is a subset of $fb(\mathcal{G})$ for any $\mathcal{G} \cap \mathcal{H} = \emptyset$, thus Lemma 39 also shows that Holant$([0,0,1,0]) \leq_T$ Holant$([0,0,1,0])[\mathcal{G}], \forall \mathcal{G} \cap \mathcal{H} = \emptyset$. Using gadget construction and polynomial interpolation, together with the characterazation of the permutable matchgate signature in [29], we can always simulate [0,0,1,0].

▶ **Lemma 40 (*).** *If $f \in \mathscr{M} - \mathscr{A}$ and is a permutable matchgate signature, then $[0,0,1,0]$ can be simulated by $\{f\}|\{[1,0],[1,0,1],[1,0,1,0]\}$ in a planar left-side manner.*

For any outside crossing graph G_{oc}, if we replace each vertex in G_{oc} with a planar gadget that simulates $[0,0,1,0]$, the obtained graph G'_{oc} is still an outside crossing graph, and consequently a ring-blowup graph and we have $G'_{oc} \in fb(\mathcal{G})$.

[4] The definition of navel in Section 5 can be seen as a special case in which $A_d = \emptyset$.



(a) (b)

Figure 5 (a) A ring-blowup of K_4. K_4 is coloured in black. (b) A visualization of G_{oc} in Lemma 39. G_{oc} has no crossing inside the dotted circle.

Furthermore, G'_{oc} can be seen as an instance of Holant$(\mathcal{F}|\widehat{\mathcal{EQ}})[\mathcal{G}]$ since $[1,0],[1,0,1],[1,0,1,0] \in \widehat{\mathcal{EQ}_{\leq 3}}$ by Lemma 19. Besides, we only introduce signatures from $\widehat{\mathcal{EQ}_{\leq 3}}$, and consequently the proof of Lemma 31 is completed.

8 Hardness for #CSP on apex graphs

In this section, we prove Lemma 30. The following theorem is our starting point.

▶ **Theorem 41** ([40]). *Counting matchings on a planar 3-regular multigraph is #P-hard.*

We first reduce the problem of counting matchings on planar 3-regular graphs to the following form:

For a 3-regular planar multigraph $G = (V, E)$, we construct $G_a = (V_a, E_a)$ where $V_a = V \cup \{a\}$ and $E_a = E \cup \{va | v \in V\}$. We also assign signatures from $(\{[0,0,0,1,0]\} \cup \widehat{\mathcal{EQ}})$ to vertices in G_a as follows: we assign a $[0,0,0,1,0]$ signature to each vertex in V, and a $[1,0,1,0,1,...,1]$ signature to a.

▶ **Lemma 42** (*). *$Z(G_a)$ is exactly the number of matchings in G.*

The following lemma completes the proof.

▶ **Lemma 43** (*). *For each signature $f \in \mathscr{M}_P - \mathscr{A}$, each $[0,0,0,1,0]$ signature assigned to a vertex in G_a can be simulated by $\{f, [1,0], [1,0,1], [1,0,1,0]\}$, such that after replacing each vertex in G with the corresponding gadget in G_a, the obtained graph G'_a is an apex graph.*

References

1 Andrei Bulatov, Martin Dyer, Leslie Ann Goldberg, Markus Jalsenius, Mark Jerrum, and David Richerby. The complexity of weighted and unweighted #CSP. *Journal of Computer and System Sciences*, 78(2):681–688, 2012. doi:10.1016/j.jcss.2011.12.002.

2 Andrei Bulatov, Martin Dyer, Leslie Ann Goldberg, Markus Jalsenius, and David Richerby. The complexity of weighted Boolean #CSP with mixed signs. *Theoretical Computer Science*, 410(38-40):3949–3961, 2009. doi:10.1016/j.tcs.2009.06.003.

3 Andrei A Bulatov. The complexity of the counting constraint satisfaction problem. *Journal of the ACM (JACM)*, 60(5):1–41, 2013. doi:10.1145/2528400.

4 Jin-Yi Cai and Xi Chen. Complexity of counting CSP with complex weights. *Journal of the ACM (JACM)*, 64(3):1–39, 2017. doi:10.1145/2822891.

5 Jin-Yi Cai, Xi Chen, and Pinyan Lu. Nonnegative weighted #CSP: an effective complexity dichotomy. *SIAM Journal on Computing*, 45(6):2177–2198, 2016. doi:10.1137/15M1032314.

6 Jin-Yi Cai and Vinay Choudhary. Valiant's Holant theorem and matchgate tensors. *Theoretical Computer Science*, 384(1):22–32, 2007. doi:10.1016/J.TCS.2007.05.015.

7 Jin-Yi Cai and Austen Z Fan. Planar 3-way edge perfect matching leads to a Holant dichotomy. *arXiv preprint arXiv:2303.16705*, 2023. doi:10.48550/arXiv.2303.16705.

8 Jin-Yi Cai and Zhiguo Fu. Holographic algorithm with matchgates is universal for planar #CSP over Boolean domain. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pages 842–855, 2017. doi:10.1145/3055399.3055405.

9 Jin-Yi Cai, Zhiguo Fu, Heng Guo, and Tyson Williams. FKT is not universal—a planar Holant dichotomy for symmetric constraints. *Theory of Computing Systems*, 66(1):143–308, 2022. doi:10.1007/S00224-021-10032-1.

10 Jin-Yi Cai, Zhiguo Fu, and Shuai Shao. New planar P-time computable six-vertex models and a complete complexity classification. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1535–1547. SIAM, 2021. doi:10.1137/1.9781611976465.93.

11 Jin-Yi Cai, Heng Guo, and Tyson Williams. A complete dichotomy rises from the capture of vanishing signatures. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pages 635–644, 2013.

12 Jin-Yi Cai and Michael Kowalczyk. Spin systems on k-regular graphs with complex edge functions. *Theoretical Computer Science*, 461:2–16, 2012. doi:10.1016/J.TCS.2012.01.021.

13 Jin-Yi Cai, Michael Kowalczyk, and Tyson Williams. Gadgets and anti-gadgets leading to a complexity dichotomy. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*, pages 452–467, 2012. doi:10.1145/2090236.2090272.

14 Jin-Yi Cai, Pinyan Lu, and Mingji Xia. The complexity of complex weighted Boolean #CSP. *Journal of Computer and System Sciences*, 80(1):217–236, 2014. doi:10.1016/J.JCSS.2013.07.003.

15 Jin-Yi Cai, Pinyan Lu, and Mingji Xia. Holographic algorithms with matchgates capture precisely tractable planar #CSP. *SIAM Journal on Computing*, 46(3):853–889, 2017. doi:10.1137/16M1073984.

16 Jin-Yi Cai and Ashwin Maran. The complexity of counting planar graph homomorphisms of domain size 3. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*, pages 1285–1297, 2023. doi:10.1145/3564246.3585173.

17 Nadia Creignou and Miki Hermann. Complexity of generalized satisfiability counting problems. *Information and computation*, 125(1):1–12, 1996. doi:10.1006/inco.1996.0016.

18 Radu Curticapean. Counting perfect matchings in graphs that exclude a single-crossing minor. *arXiv preprint arXiv:1406.4056*, 2014. arXiv:1406.4056.

19 Radu Curticapean and Mingji Xia. Parameterizing the permanent: Genus, apices, minors, evaluation mod 2k. In *2015 IEEE 56th Annual Symposium on Foundations of Computer Science*, pages 994–1009. IEEE, 2015. doi:10.1109/FOCS.2015.65.

20 Radu Curticapean and Mingji Xia. Parameterizing the permanent: Hardness for fixed excluded minors. In *Symposium on Simplicity in Algorithms (SOSA)*, pages 297–307. SIAM, 2022. doi:10.1137/1.9781611977066.23.

21 Martin Dyer, Leslie Ann Goldberg, and Mark Jerrum. The complexity of weighted Boolean #CSP. *SIAM Journal on Computing*, 38(5):1970–1986, 2009. doi:10.1137/070690201.

22 Martin Dyer and David Richerby. An effective dichotomy for the counting constraint satisfaction problem. *SIAM Journal on Computing*, 42(3):1245–1274, 2013. doi:10.1137/100811258.

23 David Eppstein and Vijay V Vazirani. NC algorithms for computing a perfect matching, the number of perfect matchings, and a maximum flow in one-crossing-minor-free graphs. In *The 31st ACM Symposium on Parallelism in Algorithms and Architectures*, pages 23–30, 2019. doi:10.1145/3323165.3323206.

24 Anna Galluccio and Martin Loebl. On the theory of Pfaffian orientations. i. perfect matchings and permanents. *the electronic journal of combinatorics*, pages R6–R6, 1999.

25 Robert Ganian, Eun Jung Kim, Friedrich Slivovsky, and Stefan Szeider. Sum-of-products with default values: Algorithms and complexity results. In *2018 IEEE 30th International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 733–737. IEEE, 2018. doi:10.1109/ICTAI.2018.00115.

26 Heng Guo and Tyson Williams. The complexity of planar Boolean #CSP with complex weights. *Journal of Computer and System Sciences*, 107:1–27, 2020. doi:10.1016/J.JCSS.2019.07.005.

27 Pieter W Kasteleyn. The statistics of dimers on a lattice: I. the number of dimer arrangements on a quadratic lattice. *Physica*, 27(12):1209–1225, 1961.

28 Boning Meng and Yicheng Pan. Dichotomies for #CSP on graphs that forbid a clique as a minor. *arXiv preprint arXiv:2504.01354*, 2025. doi:10.48550/arXiv.2504.01354.

29 Boning Meng and Yicheng Pan. Matchgate signatures under variable permutations. *arXiv preprint arXiv:2503.21194*, 2025. doi:10.48550/arXiv.2503.21194.

30 Neil Robertson and Paul D Seymour. Graph minors. XX. Wagner's conjecture. *Journal of Combinatorial Theory, Series B*, 92(2):325–357, 2004. doi:10.1016/J.JCTB.2004.08.001.

31 Neil Robertson and P.D Seymour. Graph minors. II. Algorithmic aspects of tree-width. *Journal of Algorithms*, 7(3):309–322, 1986. doi:10.1016/0196-6774(86)90023-4.

32 Simon Straub, Thomas Thierauf, and Fabian Wagner. Counting the number of perfect matchings in K_5-free graphs. *Theory of Computing Systems*, 59:416–439, 2016. doi:10.1007/S00224-015-9645-1.

33 Harold NV Temperley and Michael E Fisher. Dimer problem in statistical mechanics-an exact result. *Philosophical Magazine*, 6(68):1061–1063, 1961.

34 Glenn Tesler. Matchings in graphs on non-orientable surfaces. *Journal of Combinatorial Theory, Series B*, 78(2):198–231, 2000. doi:10.1006/JCTB.1999.1941.

35 Dimitrios M Thilikos and Sebastian Wiederrecht. Killing a vortex. In *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1069–1080. IEEE, 2022. doi:10.1109/FOCS54457.2022.00104.

36 Leslie G Valiant. The complexity of computing the permanent. *Theoretical computer science*, 8(2):189–201, 1979. doi:10.1016/0304-3975(79)90044-6.

37 Leslie G Valiant. Holographic algorithms. *SIAM Journal on Computing*, 37(5):1565–1594, 2008. doi:10.1137/070682575.

38 Vijay V Vazirani. NC algorithms for computing the number of perfect matchings in $K_{3,3}$-free graphs and related problems. *Information and computation*, 80(2):152–164, 1989. doi:10.1016/0890-5401(89)90017-5.

39 Dirk Vertigan. The computational complexity of tutte invariants for planar graphs. *SIAM Journal on Computing*, 35(3):690–712, 2005. doi:10.1137/S0097539704446797.

40 Mingji Xia, Peng Zhang, and Wenbo Zhao. Computational complexity of counting problems on 3-regular planar graphs. *Theoretical Computer Science*, 384(1):111–125, 2007. doi:10.1016/J.TCS.2007.05.023.