Sub-Exponential Time Lower Bounds for #VC and #Matching on 3-Regular Graphs

Ying Liu [1]
State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, Beijing, China
University of Chinese Academy of Sciences, Beijing, China

Shiteng Chen
State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, Beijing, China
University of Chinese Academy of Sciences, Beijing, China

Abstract

This article focuses on the sub-exponential time lower bounds for two canonical #P-hard problems: counting the vertex covers of a given graph (#VC) and counting the matchings of a given graph (#Matching), under the well-known *counting exponential time hypothesis* (#ETH).

Interpolation is an essential method to build reductions in this article and in the literature. We use the idea of block interpolation to prove that both #VC and #Matching have no $2^{o(N)}$ time deterministic algorithm, even if the given graph with N vertices is a 3-regular graph. However, when it comes to proving the lower bounds for #VC and #Matching on planar graphs, both block interpolation and polynomial interpolation do not work. We prove that, for any integer $N > 0$, we can simulate N pairwise linearly independent unary functions by gadgets with only $O(\log N)$ size in the context of #VC and #Matching. Then we use log-size gadgets in the polynomial interpolation to prove that planar #VC and planar #Matching have no $2^{o(\sqrt{\frac{N}{\log N}})}$ time deterministic algorithm. The lower bounds hold even if the given graph with N vertices is a 3-regular graph.

Based on a stronger hypothesis, *randomized exponential time hypothesis* (rETH), we can avoid using interpolation. We prove that if rETH holds, both planar #VC and planar #Matching have no $2^{o(\sqrt{N})}$ time randomized algorithm, even that the given graph with N vertices is a planar 3-regular graph. The $2^{\Omega(\sqrt{N})}$ time lower bounds are tight, since there exist $2^{O(\sqrt{N})}$ time algorithms for planar #VC and planar #Matching.

We also develop a fine-grained dichotomy for a class of counting problems, symmetric Holant*.

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

Keywords and phrases computational complexity, planar Holant, polynomial interpolation, rETH, sub-exponential, #ETH, #Matching, #VC

Digital Object Identifier 10.4230/LIPIcs.STACS.2024.49

Funding *Ying Liu* : Supported by NSFC 61932002 and NSFC 62272448.
Shiteng Chen: Supported by National Key R&D Program of China (2023YFA1009500), NSFC 61932002 and NSFC 62272448

Acknowledgements The authors are very grateful to Prof. Mingji Xia for his beneficial guidance and advice.

[1] Corresponding author

41st International Symposium on Theoretical Aspects of Computer Science (STACS 2024).
Editors: Olaf Beyersdorff, Mamadou Moustapha Kanté, Orna Kupferman, and Daniel Lokshtanov;
Article No. 49; pp. 49:1–49:18

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

1 Introduction

As an analog of NP, Valiant [19] defined the class #P of counting problems, which non-deterministic polynomial time Turing machines can compute with outputting the number of accepting computations. #P-hardness is similarly defined as NP-hardness. Two canonical counting problems, counting the vertex covers of a given graph (#VC) and counting the matchings of a given graph (#Matching), were proven to be #P-hard [18], even if the graph is sparse or planar. The two problems caused continuous attention in the past years [1, 6–8].

For the two problems, trivial $2^n\text{poly}(n)$ or $2^m\text{poly}(m)$ time algorithms exist by checking all possible solutions, where n or m denotes the number of vertices or edges in the input graph. A series of improved algorithms [9, 10, 16, 17, 22] for the two problems still need exponential time. According to the well-believed *exponential time hypothesis* (ETH) [12, 13], Dell et al. [8] put forward the counting version #ETH which states #3-SAT can not be solved in sub-exponential time. Under #ETH, many counting problems were proved that they have no $2^{o(n)}$ or $2^{o(m)}$ time deterministic algorithm. For example, counting the vertex covers of a graph with maximum degree 3 [1, 7, 15] and counting the matchings of a graph with maximum degree 4 [7] both have no $2^{o(n)}$ time algorithm. And there are also some fine-grained dichotomies [1, 15] driven from the lower bound of #VC. If the two problems are restricted on planar graphs, there are $O(2^{\sqrt{n}})$ time algorithms [16, 22]. However, the tight lower bounds for the two problems restricted on planar graphs are still open. Marx et al. [16] proved that counting the matchings of a graph, with a tree decomposition of width tw given, has no $2^{o(tw)}$ time algorithm under #ETH, even if the graph has maximum degree 8.

This article considers the lower bound results for #VC and #Matching on 3-regular graphs. We represent the two problems in the Holant framework [3], where each counting problem is defined by a set of functions. This helps us comprehend the difficulty of problems and build reductions between them more easily. One motivation behind this work is to enhance and complement the current lower bound results. Besides, the two problems on 3-regular graphs are starting problems that drive a series of dichotomy theorems [2, 4, 14]. Compared to the restriction that graphs are with maximum degree 3, the two problems on 3-regular graphs are defined by fewer functions in the Holant framework. For example, #Matching on 3-regular graphs is defined by only a ternary function in the Holant framework. This brings great convenience in building reductions. So another motivation is to prepare for developing fine-grained dichotomy theorems, which state that a problem in a class either is tractable in polynomial time or has no $2^{o(n)}$ time algorithm (or no $2^{o(\sqrt{n})}$ time algorithm on planar graphs).

We prove the $2^{\Omega(n)}$ time lower bound for #VC and #Matching on 3-regular graphs, presented in Section 3. In Section 4, we apply polynomial interpolation via log size gadgets to obtaining the nearly tight $2^{\Omega(\sqrt{\frac{n}{\log n}})}$ time lower bound for #VC and #Matching on planar 3-regular graphs under #ETH. In Section 5, we avoid the use of interpolation and prove the tight $2^{\Omega(\sqrt{n})}$ time lower bound for #VC and #Matching on planar 3-regular graphs, based on a stronger assumption rETH. In Section 6, we develop a simple fine-grained dichotomy for a class of counting problems.

2 Preliminaries

2.1 Notations and definitions

Let $\mathbb{N}$, $\mathbb{Z}$, and $\mathbb{C}$ be the set of natural numbers, the set of integers, and the set of complex numbers, respectively. $[q]$ of some positive integer q denotes the finite domain $\{1, 2, ..., q\}$. A domain of size 2 is called the Boolean domain, where any entry is assigned 0 or 1. A function

(or called signature) $F : \{0,1\}^k \to \mathbb{C}$ of some non-negative integer k is a complex-valued Boolean function, where k is called the arity of F. F is a *symmetric* function if its value is invariant under the permutation of its variables. The value of a Boolean symmetric function F only depends on the Hamming weight of the input assignment, so F can be written as $[f_0, f_1, ...f_k]$ where f_i is the value of F accepting an assignment with Hamming weight $i \in [k]$. A function of arity 1 or 2 is called a *unary function* or a *binary function*, respectively. A Boolean unary function F is usually written as a vector $[F(0), F(1)]$, and a Boolean binary function H is usually written as a matrix $\begin{pmatrix} H(0,0) & H(0,1) \\ H(1,0) & H(1,1) \end{pmatrix}$. For example, the binary equality function $=_2$ is written as $[1,0,1]$, the binary dis-equality function $\neq_2$ is written as $[0,1,0]$, the equality function $=_k$ of some arity $k \geq 3$ is written as $[1,0,...,0,1]$, the binary function OR_2 is written as $[0,1,1]$, and the ternary function OR_3 is written as $[0,1,1,1]$.

An undirected graph G is denoted by a pair (V, E), where V is the set of vertices and E is the set of edges. Multiple edges may exist between the same pair of vertices and self-loops in E. Let $N(v) \subseteq V$ and $E(v) \subseteq E$ denote the adjacent vertices and incident edges of v, respectively. The degree of a vertex v is denoted by $d_v = |E(v)|$ (a self-loop is counted twice, and an l-multiple edge of some integer l is counted l times). $\Delta = \max\{d_v \mid v \in V\}$ denotes the maximum degree of G. A bipartite graph G is also written as a tuple $(V_L \cup V_R, E)$, where V_L, V_R are two disjoint nonempty sets of vertices, and each edge $e \in E$ has one endpoint in V_L and another in V_R.

We introduce two individual counting problems on graphs in the following.

▶ **Definition 1** (#VC). *A vertex cover of a graph $G(V,E)$ is a set $S \subset V$, such that S contains at least one endpoint of e for every edge $e \in E$. The problem #Vertex Cover (#VC) is defined as*

Input: a graph $G(V,E)$,
Output: the number of vertex covers of G.

An *independent set* of a graph $G(V,E)$ is a set $S \subseteq V$, such that S contains at most one endpoint of e for every edge $e \in E$. The problem #Independent Set (#IS) is defined as counting the independent sets of a given graph. $V - S$ for any vertex cover S must be an independent set, so $\#\mathrm{IS}(G) = \#\mathrm{VC}(G)$ for any graph G with n vertices, i.e., the two problems #IS and #VC are equivalent.

▶ **Definition 2** (#Matching). *A matching of a graph $G(V,E)$ is a set $M \in E$, such that e_1 and e_2 do not intersect for any pair $e_1, e_2 \in M$. The problem #Matching is defined as*

Input: a graph $G(V,E)$,
Output: the number of matchings of G.

We use the prefix 3R-, 3Δ-, or pl- to denote the restriction that the input graphs are 3-regular, have max-degree no more than 3, or are planar, respectively. For example, #pl-3R-VC denotes counting the vertex covers for a given planar 3-regular graph.

To better analyze the complexity of the above two problems, we express them in the Boolean Holant [3] framework. A Boolean Holant problem, dubbed Holant($\mathcal{F}$), is parameterized by a set $\mathcal{F}$ of Boolean functions. A tuple $\Omega = (G, \pi)$ is called *signature grid* over $\mathcal{F}$, where $G(V,E)$ is a graph, and the mapping π assigns to every vertex $v \in V$ a function $F_v \in \mathcal{F}$ with a linear order to the edges in $E(v)$.

▶ **Definition 3** (Holant [3]). *Let $\mathcal{F}$ be a set of Boolean functions. The Boolean Holant problem* Holant($\mathcal{F}$) *is defined as*

Input: $\Omega = (G, \pi)$ *over* $\mathcal{F}$

Output: $$\text{Holant}(\Omega) = \sum_{\sigma: E \to \{0,1\}} \prod_{v \in V} F_v(\sigma|_{E(v)}),$$

where $\sigma|_{E(v)}$ is the restriction of σ to $E(v)$ and $F_v(\sigma|_{E(v)})$ depends on the ordered input tuple $\sigma|_{E(v)}$. If $\prod_{v \in V} F_v(\sigma|_{E(v)}) \neq 0$ for a fixed assignment σ, we called σ a satisfying assignment.

Given any function F and any non-zero constant λ, the two functions λF and F are equivalent in the context of Holant, since the factor λ only brings a constant multiplicative factor to the final result. The operator transforming any function F to λF for some constant $\lambda \in \mathbb{C} - \{0\}$ is called *normalization*. We usually hide the information of π into the graph G, so Ω is represented by G for simplification. If $\mathcal{F}$ is finite, then the graph G must be bounded degree. If $\mathcal{F} = \{F\}$ consists of only one function F, we directly use Holant(F) denotes Holant($\mathcal{F}$). The problem Holant*($\mathcal{F}$) denotes the problem Holant($\mathcal{F} \cup \mathcal{U}$) where $\mathcal{U}$ is the set of all unary functions.

Let $\mathcal{F}, \mathcal{H}$ be two sets of Boolean functions. A bipartite signature grid $\Omega(G, \pi)$ over $\mathcal{F}|\mathcal{H}$ consists of a bipartite graph $G(V_L \cup V_R, E)$ and a mapping π which maps each vertex $v \in V_L$ or $v \in V_R$ to a function $F \in \mathcal{F}$ or a function $H \in \mathcal{H}$, respectively. The problem Holant($\mathcal{F}|\mathcal{H}$) is similarly defined, with a bipartite signature grid Ω over $\mathcal{F}|\mathcal{H}$ as an input. Trivially, Holant($\mathcal{F}$) is equivalent to Holant($\mathcal{F}| =_2$).

We re-describe the problems #VC and #Matching as Boolean Holant problems. It is trivial to re-describe #Matching. Given a graph $G(V, E)$, we map the function $F_v = [1, 1, 0, ..., 0]$ of arity d_v to every vertex $v \in V$. The function F_v takes value 1 if no more than one incident edge of v is assigned 1; otherwise, it takes 0. Holant(G) is the number of matching of G. #Matching is exactly the problem Holant($\{[1, 1], [1, 1, 0], [1, 1, 0, 0], ...\}$), #3Δ-Matching is the problem Holant($\{[1, 1], [1, 1, 0], [1, 1, 0, 0]\}$), and #3R-Matching is the problem Holant($[1, 1, 0, 0]$).

Considering re-describing the problem #VC. Given the input graph $G(V, E)$, we map the equality function $=_{d_v}$ to each vertex $v \in V$. For every edge $e \in E$, we add a extra vertex u_e assigned the function OR_2 to divide it. These define a new bipartite signature grid $G'(V \cup V_e, E')$ where $V_e = \{u_e | e \in E\}$ denotes the set of new vertices. Given a satisfying assignment to E', let the set $S \subseteq V$ consist of the vertices $v \in V$ whose incident edges all are assigned 1. S must be a vertex cover of G. Conversely, given a vertex cover S of G, the incident edges $E'(v) \in E'$ of each $v \in S$ are assigned 1, and the other edges are assigned 0. Such an assignment must satisfy $\prod_{v \in V \cup V_e} F_v = 1$. So Holant($G'$) is the number of the vertex covers of G. #VC is exactly the problem Holant($\{=_1, =_2, ...\}|OR_2$), #3Δ-VC is the problem Holant($\{=_1, =_2, =_3\}|OR_2$), and #3R-VC is the problem Holant($=_3 |OR_2$).

2.2 Counting exponential time hypothesis

Impagliazzo et al. [12, 13] put forward the well-known *exponential time hypothesis* (ETH), which states that the satisfiability of a given 3-CNF formula (3-SAT) can not be decided in sub-exponential time. Dell et al. [8] put forward the relaxed counting version #ETH.

▶ **Conjecture 4** (#ETH [8]). *There exists a constant $\varepsilon > 0$ such that no deterministic algorithm can solve #3-SAT in $O(2^{\varepsilon n})$ time, where n denotes the number of variables of the input formula.*

The $2^{\Omega(n)}$ time lower bound can be strengthened to $2^{\Omega(m)}$ where m denotes the number of clauses in the input formula, according to the *Sparsification Lemma* [13].

A stronger hypothesis, *randomized exponential time hypothesis* (rETH) [8], states that the same lower bound also holds for the probabilistic algorithm.

▶ **Conjecture 5** (rETH [8]). *There is a constant $\varepsilon > 0$ such that no randomized algorithm can decide 3-SAT in $O(2^{\varepsilon n})$ time with error probability at most 1/3.*

Polynomial-time Turing reductions, signed as $\leq_{\text{poly}}$ or $\leq_{\text{p}}$, can not always preserve the sub-exponential time lower bound since the size of the generated instances may increase non-linearly. Impagliazoo et al. [13] introduced another particular class of Turing reductions: *sub-exponential time reduction families* (SERF), which preserves the sub-exponential time lower bound. We restrict the definition of SERF to problems on graphs.

▶ **Definition 6** ([7, 13]). *Let A and B be two problems on graphs. A sub-exponential time reduction family from A to B is an algorithm T with oracle access for B. T accepts a tuple (G, ε) as input, where G is an input of A and $\varepsilon > 0$ is a running time parameter of T, and*
(1) *computes $A(G)$ in time $O(2^{\varepsilon|V(G)|})$ with*
(2) *only invoking the oracle of B on graphs with $O(|V(G)|)$ vertices.*
If such an algorithm exists, We say that A is SERF-reducible to B, written as $A \leq_{\text{serf}} B$.

Suppose $A \leq_{\text{serf}} B$. If B has a $O(2^{\varepsilon n})$ time algorithm for some constant $\varepsilon > 0$, where n denotes the number of vertices in the input graph, then we can solve $A(G)$ in $O(2^{\varepsilon|V(G)|}) \cdot O(2^{\varepsilon \cdot O(|V(G)|)}) = O(2^{\varepsilon'|V(G)|})$ time for some constant $\varepsilon' > 0$. Conversely, if A has no sub-exponential time algorithm, so does B. SERF reductions are known to be transitive [13, Section 1.1.4].

Ying [15] built a series of SERF reductions and proved the sub-exponential time lower bound for #3Δ-VC under #ETH.

▶ **Lemma 7** ([15]). *If #ETH holds, then there exists a constant $\varepsilon > 0$ such that #3Δ-VC has no $O(2^{\varepsilon n})$ time deterministic algorithm, where n denotes the number of vertices in the input graph.*

Calabro et al. [5] proved an isolation lemma and built a SERF reduction from 3-SAT to *Unique 3-SAT*, which is a sub-problem of 3-SAT with the restriction that the input 3-CNF formula has at most one satisfying assignment.

▶ **Lemma 8** ([5]). *If rETH holds, then there exists a constant $\varepsilon > 0$ such that Unique 3-SAT has no $O(2^{\varepsilon m})$ time randomized algorithm, where m denotes the number of clauses in the input formula.*

2.3 Gadget construction

To preserve the $2^{\Omega(\sqrt{N})}$ time lower bound, the SERF reduction should be strengthened to only $2^{o(\sqrt{N})}$ time cost.

A trivial but useful method to build such a reduction is called *gadget construction*. A *gadget* is also a signature grid (G, π), where $G = (V, E \cup X)$ is a graph with some dangling edges X. A dangling edge has one endpoint in V and the other dangling. The gadget defines a function of arity $k = |X|$

$$\Gamma(x_1, x_2, ..., x_k) = \sum_{\sigma: E \to \{0,1\}} \prod_{v \in V} F_v(\hat{\sigma}|_{E(v)}),$$

where $(x_1, x_2, ..., x_k) \in \{0,1\}^k$ is an assignment to X and $\hat{\sigma}$ is the extension of the assignment σ by $(x_1, x_2, ..., x_k)$. If $F_v \in \mathcal{F}$ for every vertex $v \in V$, the gadget is called an $\mathcal{F}$-gate with signature Γ. For example, Figure 1 shows a $\{[1,1,0,0]\}$-gate with signature $[4,2,1,1]$. Given

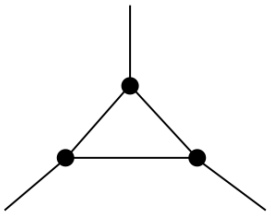


Figure 1 A $\{[1,1,0,0]\}$-gate with signature $[4,2,1,1]$.

an instance G of Holant$([4,2,1,1])$, we can replace each occurrence of $[4,2,1,1]$ by such a gadget. The value of the new instance equals the original value. So Holant$([4,2,1,1])$ is reduced to Holant$([1,1,0,0])$. Such a reduction built by *gadget construction* is a SERF reduction. Besides, the reduction only costs poly(N) time, where N denotes the number of vertices in the input graph. So the reductions built by gadget constructions can preserve the $2^{\Omega(\sqrt{N})}$ time lower bound. Ying [23] used gadget constructions to prove the following lemma.

▶ **Lemma 9** ([23]). *If #ETH holds, then there exists a constant $\varepsilon > 0$ such that planar* Holant$(\{=_2, =_3, \neq_2, OR_3\})$ *has no $O(2^{\varepsilon\sqrt{N}})$ time algorithm, where N denotes the number of vertices in the input graph.*

2.4 Holographic transformation

Another method is called *holographic transformation* [20, 21], which also preserves the $2^{\Omega(\sqrt{N})}$ time lower bound.

The tensor product $\otimes$ is the Kronecker product, that is, for two matrices $X = X_{a\times b}$ and $Y = Y_{c\times d}$, $X \otimes Y$ is an $ac \times bd$ matrix with entry $X_{i,j}Y_{k,l}$ at $(i,k) \in [a] \times [c]$ row and $(j,l) \in [b] \times [d]$ column, where a, b, c, d are some positive integers. Tensor power is defined recursively $X^{\otimes k} = X^{\otimes(k-1)} \otimes X$ for some integer $k > 0$.

Let T be a 2×2 invertible matrix. Given a Boolean function F of some arity k, written as a column vector in $\mathbb{C}^{2^k}$, we write $TF = T^{\otimes k}F$ as the transformed function. For a set $\mathcal{F}$ of functions, $T\mathcal{F} = \{TF | F \in \mathcal{F}\}$. $FT = FT^{\otimes k}$ and $\mathcal{F}T = \{FT | F \in \mathcal{F}\}$ are similarly defined, where F is written as a row vector. Given an instance G of Holant$(\mathcal{F}|\mathcal{H})$, we generate a new bipartite graph G' from G, by reassigning the function FT or $T^{-1}H$ to each vertex which is assigned the function $F \in \mathcal{F}$ or $H \in \mathcal{H}$, respectively. Valiant's Holant Theorem [21] shows that Holant(G) = Holant(G').

▶ **Lemma 10** ([21]). *Let $\mathcal{F}$ and $\mathcal{H}$ be two function sets. Given an invertible 2×2 matrix T,*

$$\text{Holant}(\mathcal{F}|\mathcal{H}) \leq_{\text{p}} \text{Holant}(\mathcal{F}T|T^{-1}\mathcal{H}).$$

In addition, the generated instance has the same number of vertices and edges as the original.

3 Lower bounds for #3R-VC and #3R-Matching under #ETH

In this section, we build SERF-reductions from #3Δ-VC to prove the $2^{\Omega(N)}$ time lower bounds for #3R-VC and #3R-Matching, where N denotes the number of vertices in the input graph.

3.1 Lower bound for #3R-VC

We build a SERF-reduction from #3Δ-VC to #3R-VC. In fact, we prove the equivalent reduction Holant$(\{=_1,=_2,=_3\}|OR_2) \leq_{\text{serf}}$ Holant$(=_3\ |OR_2)$ by gadget constructions.

▶ **Theorem 11.** *If #ETH holds, then there exists a constant $\varepsilon > 0$ such that the number of vertex covers of a 3-regular graph with N vertices can not be computed in $O(2^{\varepsilon N})$ time.*

Proof. We first reduce Holant$(\{=_1,=_2,=_3\}|OR_2)$ to Holant$(=_3\ |\{OR_2,[1,1]\})$. A function $=_1$ can be realized by one function $=_3$ connected with two unary functions $[1,1]$, and a function $=_2$ can be realized by one function $=_3$ connected with one unary function $[1,1]$. We realize $[1,1]$ by the $\{=_3\}|\{OR_2\}$-gate shown in Figure 2.

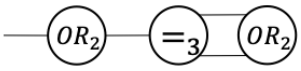


Figure 2 A $\{=_3\}|\{OR_2\}$ gate with signature $[1,1]$.

By the above, Holant$(\{=_1,=_2,=_3\}|OR_2) \leq_{\text{serf}}$ Holant$(=_3\ |OR_2)$ and we can conclude this theorem, according to Lemma 7. ◀

3.2 Lower bound for #3R-Matching

We build a SERF-reduction from #3R-VC to #3R-Matching, i.e., we demonstrate that Holant$(OR_2|\ =_3) \leq_{\text{serf}}$ Holant$([1,1,0,0])$ by the idea of *block interpolation* [7], which is actually multivariate polynomial interpolation.

We first introduce a lemma, which is essential during the interpolation process.

▶ **Lemma 12** ([11, 18]). *Let A,B,C,D be positive rational numbers, and x_0,y_0 be rational numbers. Define the sequences $\{x_l\}_{l\geq 0}$ and $\{y_l\}_{l\geq 0}$ recursively by $x_{l+1} = Ax_l + By_l$ and $y_{l+1} = Cx_l + Dy_l$. Then the sequence $\{\frac{x_l}{y_l}\}_{l\geq 0}$ is pairwise different as long as $AD - BC \neq 0$ and $By_0^2 - Cx_0^2 - (A-D)x_0y_0 \neq 0$.*

▶ **Theorem 13.** *If #ETH holds, then there exists some constant $\varepsilon > 0$ such that #3R-Matching can not be calculated in $O(2^{\varepsilon N})$ time, where N is the number of vertices in the input graph.*

Proof. We establish the following reduction chain.

$$\begin{aligned}
\text{Holant}([0,1,1]\mid[1,0,0,1]) &\leq_{\text{serf}} \text{Holant}([-1,2,0]\mid[4,2,1,1]) && (1)\\
&\leq_{\text{serf}} \text{Holant}(\{[-1,2,0],[1,1,0,0]\}) && (2)\\
&\leq_{\text{serf}} \text{Holant}([1,1,0,0]) && (3)
\end{aligned}$$

1. The reduction (1) is proved by the holographic transformation defined by $Q = \begin{pmatrix} 0 & \sqrt[3]{4} \\ \frac{1}{2}\sqrt[3]{4} & \frac{1}{2}\sqrt[3]{4} \end{pmatrix}$, since

$$[0,1,1](Q^{-1})^{\otimes 2} = (0,1,1,1)(\frac{1}{\sqrt[3]{2}}\begin{pmatrix}\frac{1}{2}\sqrt[3]{4} & -\sqrt[3]{4}\\ -\frac{1}{2}\sqrt[3]{4} & 0\end{pmatrix} \otimes \frac{1}{\sqrt[3]{2}}\begin{pmatrix}\frac{1}{2}\sqrt[3]{4} & -\sqrt[3]{4}\\ -\frac{1}{2}\sqrt[3]{4} & 0\end{pmatrix})$$
$$= (0,1,1,1)\frac{\sqrt[3]{4}}{4}\begin{pmatrix}1 & -2 & -2 & 4\\ -1 & 0 & 2 & 0\\ -1 & 2 & 0 & 0\\ 1 & 0 & 0 & 0\end{pmatrix}$$
$$= \frac{\sqrt[3]{4}}{4}(-1,2,2,0) = \frac{\sqrt[3]{4}}{4}[-1,2,0]$$

and

$$Q^{\otimes 3}[1,0,0,1] = (\frac{\sqrt[3]{4}}{2})^3\begin{pmatrix}0&0&0&0&0&0&0&8\\0&0&0&0&0&0&4&4\\0&0&0&0&0&4&0&4\\0&0&0&0&2&2&2&2\\0&0&0&4&0&0&0&4\\0&0&2&2&0&0&2&2\\0&2&0&2&0&2&0&2\\1&1&1&1&1&1&1&1\end{pmatrix}\begin{pmatrix}1\\0\\0\\0\\0\\0\\0\\1\end{pmatrix} = \frac{1}{2}\begin{pmatrix}8\\4\\4\\2\\4\\2\\2\\2\end{pmatrix} = \begin{pmatrix}4\\2\\2\\1\\2\\1\\1\\1\end{pmatrix}.$$

$(4,2,2,1,2,1,1,1)^T$ has the indexes 000, 001, 010, 011,100,101,110, and 111 in order, and it is the column vector of the function $[4,2,1,1]$.

2. The reduction (2) is built by gadget constructions. We replace each occurrence of the function $[4,2,1,1]$ by the $\{[1,1,0,0]\}$-gate showed in Figure-1.

3. The reduction (3) is built by the idea of block interpolation. Let G be an instance of Holant$(\{[-1,2,0],[1,1,0,0]\})$ and $V' \subseteq V(G)$ denote the vertices assigned the function $[-1,2,0]$. Suppose $|V'| = n \leq |V(G)|$. We divide V' to $r = \frac{n}{d}$ disjoint blocks $B_1, B_2, ..., B_r$ for some positive integer d, such that each block $|B_i| = d$ for $i \in [r]$ (w.l.o.g, n is divisible by d.)[2]. We label each satisfying assignment to $E(G)$ a type $\vec{t} = (t_1, t_2, ..., t_r) \in \{0,1,...,d\}^r$, where t_i denotes the number of vertices $v \in B_i$ with $E(v)$ assigned $(0,0)$. Let $\rho_{\vec{t}}$ denote the number of satisfying assignments with type $\vec{t}$. Then Holant$(G) = \sum_{\vec{t}} \rho_{\vec{t}} \prod_{i=1}^{r} (-1)^{t_i}(2)^{d-t_i}$. Define a multivariate polynomial

$$\mu(x_1, x_2, ..., x_r) = \sum_{\vec{t}} \rho_{\vec{t}} \prod_{i=1}^{r} (x_i)^{t_i}$$

on variables $x_1, x_2, ..., x_r \in \mathbb{C}$. Holant$(G) = 2^n \cdot \mu(-\frac{1}{2}, -\frac{1}{2}, ..., -\frac{1}{2})$.
Given $\vec{l} = (l_1, l_2, ..., l_r) \in \mathbb{N}^r$, we replace each vertex $v \in B_i$ by a gadget with a binary signature which is realized by a ternary function $[1,1,0,0]$ connecting with a unary function showed in Figure 3.
The unary function realized by the gadget S_l can be written as $[s_l^0, s_l^1]$ where $\{s_l^0\}$ and $\{s_l^1\}$ satisfy the recurrences: $s_l^0 = 25s_{l-1}^0 + 13s_{l-1}^1$ and $s_l^1 = 13s_{l-1}^0 + 7s_{l-1}^1$ with initial conditions $s_0^0 = 50$ and $s_0^1 = 26$. Actually, s_l^0 represents the number of matchings of the underlying graph of S_l, which do not include the dangling edge. And s_l^1 represents the number of matchings that include the dangling edge. If we connect $[s_l^0, s_l^1]$ to a ternary function $[1,1,0,0]$, then we realize a binary function $[s_l^0 + s_l^1, s_l^0, 0]$. The value of $G_{\vec{l}}$ is

$$\text{Holant}(G_{\vec{l}}) = \sum_{\vec{t}} \rho_{\vec{t}} \prod_{i=1}^{r} (s_{l_i}^0 + s_{l_i}^1)^{t_i}(s_{l_i}^0)^{d-t_i} = \prod_{i=1}^{r} (s_{l_i}^0)^d \cdot \mu(1 + \frac{s_{l_1}^1}{s_{l_1}^0}, 1 + \frac{s_{l_2}^1}{s_{l_2}^0}, ..., 1 + \frac{s_{l_r}^1}{s_{l_r}^0}).$$

[2] We can add the gadget, an isolated 3-multiple edge whose two endpoints are assigned $[1,1,0,0]$ and one edge is divided by an extra vertex assigned $[-1,2,0]$, to fit the assumption without affecting Holant(G).

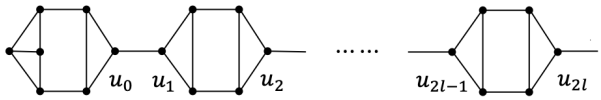


Figure 3 A $\{[1,1,0,0]\}$-gate S_l for some integer l.

We construct a series of graphs $G_{\vec{l}}$ for $\vec{l} \in [d+1]^r$ to invoke the oracle of Holant$([1,1,0,0])$. Then, we obtain the values of μ in $(d+1)^r$ distinct points, since $\{\frac{s_l^1}{s_l^0}\}_{l\geq 0}$ is pairwise different according to Lemma 12. So we can solve all coefficients $\rho_{\vec{t}}$ of μ by Lagrange interpolation, and compute Holant(G) in poly$((d+1)^r)$ time. Each $G_{\vec{l}}$ is constructed in poly$(|V(G)|)$ time and it has $O(d|V(G)|)$ vertices.
Give a running time parameter ε, we choose a constant d such that $\frac{\log(d+1)}{d} \leq \varepsilon$. The total time of the above reduction is

$$(d+1)^r(\text{poly}(|V(G)|)) + \text{poly}((d+1)^r) = O(2^{\frac{\log(d+1)}{d}n}) = O(2^{\varepsilon|V(G)|}).$$

The above reduction is a SERF reduction.

SERF reductions are transitive. So Holant$(OR_2|\ =_3) \leq_{\text{serf}}$ Holant$([1,1,0,0])$. This lemma is true by Theorem 11. ◀

4 Lower bounds for #pl-3R-VC and #pl-3R-Matching under #ETH

We build reductions from pl-Holant$(\{=_2, =_3, \neq_2, OR_3\})$ which has the $2^{\Omega(\sqrt{N})}$ time lower bound. Gadget constructions and holographic transformations preserve the $2^{\Omega(\sqrt{N})}$ time lower bound, but their ability is limited in building reductions. We need a more potent method: interpolation.

Unfortunately, both polynomial interpolation and block interpolation can not build a reduction preserving the $2^{\Omega(\sqrt{N})}$ time lower bound. The reduction built by polynomial interpolation costs polynomial time but generates new graphs with $O(N^2)$ size. The reduction built by block interpolation generates new graphs with $O(N)$ size but costs $2^{o(N)}$ time. In the process of block interpolation, the proof of Theorem 13 as an example, we can choose $d = O(\sqrt{N \log N})$ such that the time costs is $2^{o(\sqrt{N})}$, but the generated graphs have $O(N\sqrt{N \log N})$ size.

The type of interpolations that preserves the $2^{\Omega(\sqrt{N})}$ time lower bound has yet to be developed. We struggled for it but failed. We step back and consider building reductions that cost $2^{o(\sqrt{N})}$ time and generate graphs with $O(N \log N)$ vertices.

4.1 Polynomial interpolation via log size gadgets

In the traditional application of interpolation, for any integer $d > 0$, people build a series of $O(d)$ size gadgets to realize a sequence of d pairwise linearly independent functions and generate new instances with size $O(dN)$. Inspired by the proof of Theorem 1.3 in [8], we innovatively put up the way to construct a sequence of d pairwise linearly independent functions by gadgets only with size $O(\log d)$.

▶ **Lemma 14.** *Let $A, B \in \mathbb{C}^{2\times 2}$ be two non-singular matrices.*

Given a nonzero column vector $s \in \mathbb{C}^2$, the sequence of column vectors $\{M_i \cdots M_1 s\}_{i\geq 1}$ with $M_i \in \{A, B\}$ is pairwise linearly independent if the following conditions are satisfied.

(1) $\det([As\ Bs]) \neq 0$, *and*

(2) *for any pair column vectors $v_1, v_2 \in \{M_i \cdots M_1 s\}_{i\geq 1}$ (v_1, v_2 are not necessarily distinct),* $\det([Av_1\ Bv_2]) \neq 0$.

Proof. We prove by induction on the positive integer i.

1. $i = 1$. As and Bs are linearly independent according to the condition (1).
2. Inductively we assume the lemma is proven for $i = l - 1$ for any $l \geq 2$, and now assume $i = l$. Let v_1, v_2 be two distinct column vectors in $\{M_{l-1} \times \cdots \times M_1 s\}$. v_1, v_2 are linearly independent. Av_1, Av_2 are linearly independent, otherwise, $|A| = 0$ or v_1, v_2 are linearly dependent, which contradicts to the assumption. So the sequence $\{AM_{l-1} \cdots M_1 s\}$ is pairwise linearly independent. Similarly, the sequence $\{B \times M_{l-1} \cdots M_1 s\}$ is pairwise linearly independent.

   Av_1, Bv_2 for any pair column vectors in $\{M_{l-1} \times \cdots \times M_1 s\}$ are linearly independent according to the condition (2). So The sequence $\{AM_{l-1} \cdots M_1 s, BM_{l-1} \cdots M_1 s\}$ is pairwise linearly independent.

This lemma is true. ◀

If we have three gadgets with signatures A, B, s which satisfy the conditions in Lemma 14, then, for any $d \in \mathbb{N}$, we can construct a sequence of d gadgets with unary signatures which are pairwise linearly independent. Besides, each gadget has $O(\log d)$ size.

▶ **Lemma 15.** *Let $\mathcal{F}$ be a set of complex-valued Boolean functions. Suppose the following gadgets can be realized by the $\mathcal{F}$.*

(1) *two non-singular recursive gadgets with binary signature $A, B \in \mathbb{C}^{2\times 2}$ and*

(2) *a unary start gadget with signature s written as a column vector,*

which satisfy $\det([As\ Bs]) \neq 0$ *and* $\det([Av_1\ Bv_2]) \neq_2 0$ *for any unary signatures $v_1, v_2 \in \{M_i \cdots M_1 s\}_{i\geq 0}$ with $M_i \in \{A, B\}$.*

Then for a finite sequence of unary functions $\mathcal{S} = \{[x_1, y_1], \cdots, [x_m, y_m]\}$ with $x_j, y_j \in \mathbb{C}$ for any $j \in [m]$, $\text{Holant}(\mathcal{F} \cup \mathcal{S}) \leq_{\text{p}} \text{Holant}(\mathcal{F})$. *Furthermore,* $\text{Holant}(\mathcal{F})$ *has no* $2^{o(\sqrt{\frac{N}{\log N}})}$ *time algorithm if* $\text{Holant}(\mathcal{F} \cup \mathcal{S})$ *has no* $2^{o(\sqrt{n})}$ *time algorithm, where N, n denote the number of vertices of the input.*

The result also holds for planar $\text{Holant}(\mathcal{F})$ *if the gadgets are planar.*

Proof. Let $G(V \cup S, E)$ with n vertices be an instance of $\text{Holant}(\mathcal{F} \cup \mathcal{S})$, where S denotes the set of vertices each assigned a unary function in $\mathcal{S}$. Let $S_j \subseteq S$ denotes the set of vertices assigned $[x_j, y_j]$ where $j \in [m]$. We label each assignment to E a type $t = (t_1, t_2, ..., t_m)$ where $t_j \in \{0, 1, 2, ..., |S_j|\}$ denotes the number of vertices $v \in S_j$ whose incident edge is assigned 0. Then $\text{Holant}(G) = \sum_t \rho_t \prod_{j\in[m]} (x_j)^{t_j} (y_j)^{|S_j|-t_j}$ where ρ_t denotes the sum of the products of the signatures in V under the assignments with type t.

If we obtain the values of all coefficients ρ_t, then we can compute $\text{Holant}(G)$. According to Lemma 14, for any integer $n > 0$, we can simulate a sequence of n pairwise linearly independent unary gadgets $\{[w_1, z_1], [w_2, z_2], ..., [w_n, z_n]\}$ by the signature set $\mathcal{F}$. Each gadget has $O(\log n)$ size.

Define $l = (l_1, l_2, ..., l_m)$ where $l_j \in \mathbb{N}$ for $j \in [m]$. We construct a graph G_l by replacing each vertex in S_j with a vertex assigned the signature $[w_{l_j}, z_{l_j}]$. The value of G_l is

$$\text{Holant}(G_l) = \sum_t \rho_t \prod_{j\in[m]} (w_{l_j})^{t_j} (z_{l_j})^{|S_j|-t_j} \tag{1}$$

By taking $l \in [|S_1|+1] \times [|S_2|+1] \times \cdots \times [|S_m|+1]$, we construct a system of $(|S_1|+1) \times (|S_2|+1) \times \cdots \times (|S_m|+1) \leq (n+1)^m$ equations of the form as (1):

$$\begin{pmatrix} \text{Holant}(G_{(1,1,\ldots,1)}) \\ \text{Holant}(G_{(1,1,\ldots,2)}) \\ \vdots \\ \text{Holant}(G_{(|S_1|+1,|S_2|+1,\ldots,|S_m|+1)}) \end{pmatrix} = M \cdot \begin{pmatrix} \rho_{(0,0,\ldots,0)} \\ \rho_{(0,0,\ldots,1)} \\ \vdots \\ \rho_{(|S_1|,|S_2|,\ldots,|S_m|)} \end{pmatrix}$$

where the matrix $M = M_1 \otimes M_2 \otimes \cdots \otimes M_m$ with

$$M_j = \begin{pmatrix} z_1^{|S_j|} & w_1 z_1^{|S_j|-1} & \cdots & w_1^{|S_j|} \\ z_2^{|S_j|} & w_2 z_2^{|S_j|-1} & \cdots & w_2^{|S_j|} \\ \vdots & \vdots & \ddots & \vdots \\ z_{|S_j|+1}^{|S_j|} & w_{|S_j|+1} z_{|S_j|+1}^{|S_j|-1} & \cdots & w_{|S_j|+1}^{|S_j|} \end{pmatrix}$$

for $j \in [m]$. Since $\{[w_1, z_1], ..., [w_{|S_j|+1}, z_{|S_j|+1}]\}$ is pairwise linearly independent, $\det(M_j) \neq 0$ for any $j \in [m]$. So the matrix M is invertible.

We can compute all ρ_t by solving the system after obtaining the values of $\text{Holant}(G_l)$ by invoking the oracle of $\text{Holant}(\mathcal{F})$. Let $T(N)$ be the time cost of the oracle, where N is the vertices number of the input graph. In the above reduction, we build at most $(n+1)^m$ new graphs in $(n+1)^m \text{poly}(n)$ time, and each new graph has $O(n \log(n+1))$ vertices. The reduction time is $(n+1)^m \text{poly}(n) + (n+1)^m T(c \cdot n \log(n+1)) + \text{poly}((n+1)^m)$ where c is some constant. Since m is also a constant, the reduction is a polynomial time reduction. So $\text{Holant}(\mathcal{F} \cup \mathcal{S}) \leq_{\text{p}} \text{Holant}(\mathcal{F})$.

Suppose $\text{Holant}(\mathcal{F})$ has $2^{o(\sqrt{\frac{N}{\log N}})}$ time algorithm, that is, $T(N) = 2^{\varepsilon(\sqrt{\frac{N}{\log N}})}$ for any $\varepsilon > 0$. Then we can solve $\text{Holant}(G)$ in

$$(n+1)^m \text{poly}(n) + (n+1)^m 2^{\varepsilon(\sqrt{\frac{cn\log(n+1)}{\log c + \log n + \log\log(n+1)}})} + \text{poly}((n+1)^m) \leq 2^{\varepsilon'\sqrt{n}} \tag{2}$$

for any $\varepsilon' > 0$, by choose small enough ε. Then we obtain the $2^{o(\sqrt{n})}$ algorithm for the problem $\text{Holant}(\mathcal{F} \cup \mathcal{S})$. It is a contradiction to the assumption that $\text{Holant}(\mathcal{F} \cup \mathcal{S})$ has no $2^{o(\sqrt{n})}$ time algorithm. ◀

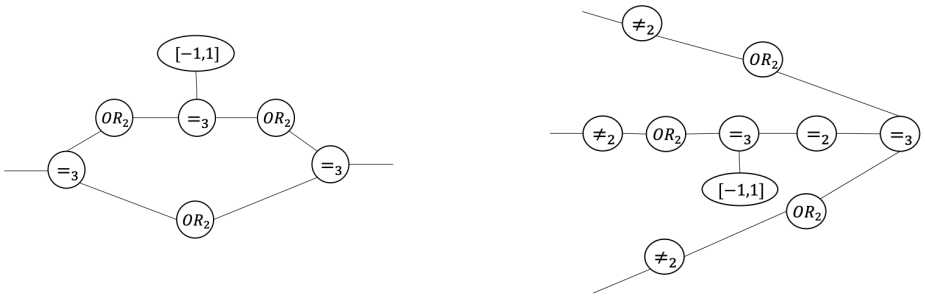
Using block interpolation via log size gadget does not improve the lower bound.

4.2 Lower bound for planar #3R-VC

We reduce pl-Holant($\{=_2, =_3, \neq_2, OR_3\}$) to pl-Holant($=_3 | OR_2$). We use the problem pl-Holant($=_3 | \{OR_2, [-1, 1]\}$) as an intermediate.

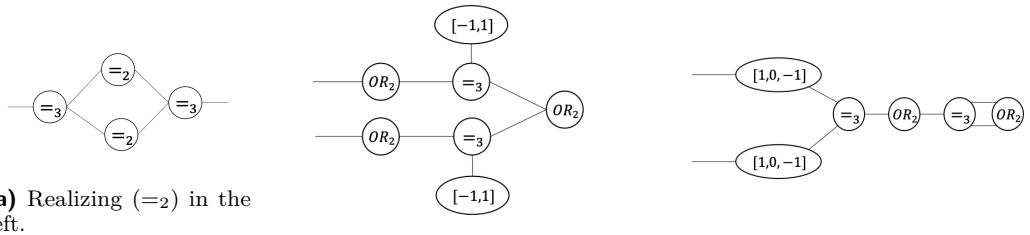
▶ **Lemma 16.** *If #ETH holds, then there exists some constant $\varepsilon > 0$ such that planar* $\text{Holant}(=_3 | \{OR_2, [-1, 1]\})$ *has no $O(2^{\varepsilon\sqrt{N}})$ time algorithm, where N denotes the number of vertices of the instance.*

Proof. We prove that pl-Holant($\{=_2, =_3, \neq_2, OR_3\}$) $\leq_{\text{p}}$ pl-Holant($=_3 | \{OR_2, [-1, 1]\}$) by gadget constructions. For any instance G of pl-Holant($\{=_2, =_3, \neq_2, OR_3\}$), we add a vertex assigned the function $=_2$ to divide each edge, and replace every occurrence of $\neq_2$ or OR_3 by the corresponding $\{=_3\}|\{OR_2, =_2, [-1, 1]\}$-gates shown in Figure 4. We generate a new graph G' which is an instance of pl-Holant($\{=_2, =_3\}|\{OR_2, [-1, 1], =_2\}$).



(a) Realizing $\neq_2$. **(b)** Realizing OR_3.

Figure 4 The constructions of some gadgets with signature $\neq_2$ and OR_3.



(a) Realizing $(=_2)$ in the left. **(b)** Realizing $[1, 0, -1]$. **(c)** Realizing $(=_2)$ in the right.

Figure 5 The constructions of gadgets with signature $(=_2)$.

Then we replace each the occurrence of $=_2$ in the left of G' by a $\{=_3\}|\{=_2\}$-gate shown in Figure 5-(a) and each occurrence of $=_2$ in the right of G' by the $\{=_3\}|\{OR_2, [-1, 1]\}$-gate shown in Figure 5-(c). The finial graph G'' is an instance of Holant$(=_3 |\{OR_2, [-1, 1]\})$.

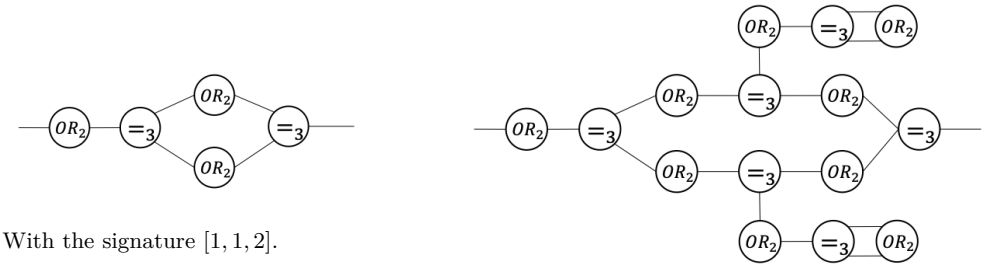
So Holant$(\{=_2, =_3, \neq_2, OR_3\}) \leq_{\mathrm{p}}$ Holant$(=_3 |\{OR_2, [-1, 1]\})$ with preserving planarity. The reductions cost polynomial time, and G'' has cN vertices for some constant c, where N denotes the number of vertices in G. Suppose Holant(G'') can be solved in $O(2^{\varepsilon\sqrt{cN}})$ time for any $\varepsilon > 0$, then Holant(G) can be solved in poly$(N) + O(2^{\varepsilon\sqrt{cN}}) = O(2^{\varepsilon'\sqrt{N}})$ time for any $\varepsilon' > 0$. It is a contradiction to Lemma 9. ◀

Next we interpolate the function $[-1, 1]$ in the context of planar Holant$(=_3 |OR_2)$.

▶ **Theorem 17.** *If #ETH holds, then there exists some constant $\varepsilon > 0$ such that planar* Holant$(=_3 |OR_2)$ *has no $O(2^{\varepsilon\sqrt{\frac{N}{\log N}}})$ time algorithm, where N denotes the number of vertices of the instance.*

Proof. In the context of planar Holant$(=_3 |OR_2)$, we build two recursive gadgets with signatures $A = \begin{pmatrix} 1 & 1 \\ 1 & 2 \end{pmatrix}, B = \begin{pmatrix} 1 & 4 \\ 2 & 5 \end{pmatrix}$ shown in Figure 6, and a start gadget with signature $s = [1, 1]^T$ shown in Figure 2. $\det(A), \det(B) \neq 0$ and $\det([As\ Bs]) \neq 0$.

Consider the sequence $\{M_i \cdots M_1 s\}_{i \geq 1}$ where $M_i \in \{A, B\}$. The elements in A, B, s are positive, so any unary signature in this sequence is written as $[z, w]$ with $z, w > 0$ and can be normalized as $[x, 1]$ with $0 < x \leq 1$. Let $v_1 = [x_1, 1]^T, v_2 = [x_2, 1]^T$ be two unary functions in the sequence. If $\det([Av_1\ Bv_2]) = 0$ then $x_1x_2 + x_1 = 3$, that is a contradiction since $0 < x_1, x_2 \leq 1$. So, the conditions in Lemma 15 are satisfied. According to Lemma 15 and Lemma 16, this theorem is true. ◀



(a) With the signature $[1, 1, 2]$.

(b) With the signature $\begin{pmatrix} 1 & 4 \\ 2 & 5 \end{pmatrix}$.

Figure 6 The recursive gadgets to simulate a sequence of unary functions of the form $\{[x_i, 1]\}_{i\geq 0}$ in the context of planar Holant$(=_3 |OR_2)$.

4.3 Lower bound for planar #3R-Matching

We reduce from planar Holant$(=_3 |\{OR_2, [-1, 1]\})$. We firstly do a holographic transformation defined by $Q = \begin{pmatrix} \frac{\sqrt[3]{4}}{2} & -\frac{\sqrt[3]{4}}{2} \\ -\sqrt[3]{4} & 0 \end{pmatrix}$. $Q^{\otimes 2}(0, 1, 1, 1)^T = \frac{(\sqrt[3]{4})^2}{4}(-1, 2, 2, 0)^T$, $Q(-1, 1)^T = \sqrt[3]{4}(-1, 1)^T$, and $(=_3)(Q^{-1})^{\otimes 3} = [4, 2, 1, 1]$. Then pl-Holant$(=_3 |\{OR_2, [-1, 1]\})$ is reduced to pl-Holant$([4, 2, 1, 1]|\{[-1, 2, 0], [-1, 1]\})$. The function $[4, 2, 1, 1]$ can be realized by a $\{[1, 1, 0, 0]\}$-gate showed in Figure 3 and the function $[-1, 2, 0]$ can be realized by the function $[1, 1, 0, 0]$ connected with a unary function $[2, -3]$, so pl-Holant$([4, 2, 1, 1]|\{[-1, 2, 0], [-1, 1]\})$ can be reduce to pl-Holant$(\{[1, 1, 0, 0], [2, -3], [-1, 1]\})$ with preserving the $2^{\Omega(\sqrt{N})}$ time lower bound.

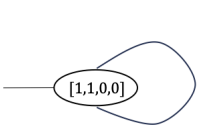
▶ **Lemma 18.** *If #ETH holds, then there exists some constant $\varepsilon > 0$ such that planar* Holant$(\{[1, 1, 0, 0], [2, -3], [-1, 1]\})$ *has no $O(2^{\varepsilon\sqrt{N}})$ time algorithm, where N is the number of vertices in the input.*

Then we interpolate the two unary functions in the context of planar Holant$(\{[1, 1, 0, 0]\})$.

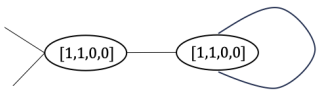
▶ **Theorem 19.** *If #ETH holds, then there exists some constant $\varepsilon > 0$ such that planar* Holant$([1, 1, 0, 0])$ *has no $O(2^{\varepsilon\sqrt{\frac{N}{\log N}}})$ time algorithm, where N is the number of vertices in the input.*

Proof. In the context of planar Holant$([1, 1, 0, 0])$, we construct two recursive gadgets $A = \begin{pmatrix} 2 & 1 \\ 1 & 0 \end{pmatrix}, B = \begin{pmatrix} 6 & 3 \\ 3 & 2 \end{pmatrix}$ and a start gadget $s = [1, 1]^T$, showed in Figure 7. $\det(A), \det(B) \neq 0$ and $\det([As\ Bs]) \neq 0$.

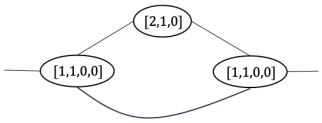
Consider the sequence $\{M_i \cdots M_1 s\}_{i\geq 1}$ where $M_i \in \{A, B\}$. Each unary function in the sequence can be normalized to $[x, 1]$ with $x > 0$. Let $v_1 = [x_1, 1]^T, v_2 = [x_2, 1]^T$ be two unary signatures in the sequence. If $\det([Av_1\ Bv_2]) = 0$ then $x_1 + 3x_2 + 2 = 0$, which is a contradiction. So this lemma is true by Lemma 15 and Lemma 18. ◀



(a) Start gadget with signature $[1, 1]$.



(b) Recursive gadget with signature $\begin{pmatrix} 2 & 1 \\ 1 & 0 \end{pmatrix}$.



(c) Recursive gadget with signature $\begin{pmatrix} 6 & 3 \\ 3 & 2 \end{pmatrix}$ where $[2, 1, 0]$ is realized by (b).

Figure 7 The gadgets to simulate a sequence of unary functions of the form $\{[x_i, 1]\}_{i\geq 0}$ in the context of planar Holant($\{[1, 1, 0, 0]\}$).

5 Lower bounds for #pl-3R-VC and #pl-3R-Matching under rETH

It can be observed that the use of interpolation weakens the lower bounds with the $\sqrt{\log N}$ factor in the exponent. In this section, based on a stronger assumption rETH, we build polynomial reductions from Unique 3-SAT. We use the character that, for any instance G of Unique 3-SAT, Holant(G) is either 0 or 1. In these reductions, we do not require the exact values of the generated instances; we only need to decide whether the values are 0 or not. Therefore, we can avoid the use of interpolation.

Unique 3-SAT can also be treated as a sub-problem of #3-SAT that is equivalent to Holant$(\{=_1, =_2, ...\} \cup \{\neq_2, OR_1, OR_2, OR_3\})$, where $OR_1 = [0, 1]$. According to the reductions in [23], Holant$(\{=_1, =_2, ...\} \cup \{\neq_2, OR_1, OR_2, OR_3\})$ can be reduced to pl-Holant$(\{=_2, =_3, \neq_2, OR_3\})$ in polynomial time by equivalent gadget constructions. That is, given a 3-CNF formula ϕ with N variables and M clauses, we can transform ϕ to a planar graph G_ϕ with $O((M + N)^2) = O(M^2)$ vertices in poly$((M + N)^2) =$ poly(M^2) time. And G_ϕ is an instance of pl-Holant$(\{=_2, =_3, \neq_2, OR_3\})$. Suppose ϕ has at most one satisfying assignment, that is, #SAT(ϕ) = Holant(G_ϕ) is either 0 or 1.

In the following, we consider transforming G_ϕ to some instance of #pl-3R-VC or #pl-3R-Matching.

5.1 Lower bound for planar #3R-VC

According to the proof of Lemma 16, we transfer G_ϕ to a graph G which is an instance of pl-Holant$(=_3 |\{OR_2, [-1, 1]\})$. G has $O(M^2)$ vertices and Holant(G) = Holant(G_ϕ) is either 0 or 1.

Let G' be a graph constructed from G by replacing each occurrence of $[1, -1]$ by a gadget with signature $[1, 1]$, showed in Figure 2. Since $1 \equiv -1 \mod 2$, (Holant$(G') \mod 2) =$ Holant(G). G' is an instance of planar Holant$(=_3 |OR_2)$. Suppose G has cM^2 vertices for some constant c. If we can compute Holant(G') in $2^{o(\sqrt{cM^2})}$ time, i.e., in $2^{\varepsilon\sqrt{cM^2}}$ time for any $\varepsilon > 0$, then we can compute #SAT(ϕ) by the above in $2^{\varepsilon\sqrt{cM^2}} +$ poly$(M^2) \leq 2^{\varepsilon' M}$ time for any $\varepsilon' > 0$. It contradicts Lemma 8. So we can obtain the following lemma.

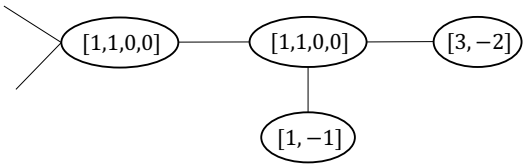
▶ **Theorem 20.** *If rETH holds, then there exists some constant $\varepsilon > 0$ such that planar* Holant$(=_3 |OR_2)$*, i.e., counting the vertex covers of a given planar 3-regular graph, has no $O(2^{\varepsilon\sqrt{N}})$ time randomized algorithm. N denotes the number of vertices of the input graph.*

5.2 Lower bound for planar #3R-Matching

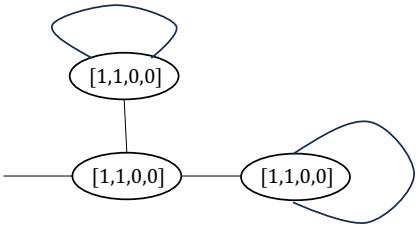
We obtain the graph G from G_ϕ as in Section 5.1. Then we do a holographic transformation defined by $Q = \begin{pmatrix} \frac{\sqrt[3]{4}}{2} & -\frac{\sqrt[3]{4}}{2} \\ -\sqrt[3]{4} & 0 \end{pmatrix}$. Because $(=_3)(Q^{-1})^{\otimes 3} = [4,2,1,1]$, $Q^{\otimes 2}(0,1,1,1)^T = (\sqrt[3]{4})^{-1}(-1,2,2,0)^T$ or $Q(-1,1)^T = \sqrt[3]{4}(-1,1)^T$ The generated instance G_1 is an instance of Holant$([4,2,1,1]|\{(\sqrt[3]{4})^{-1}[-1,2,0], \sqrt[3]{4}[-1,1]\})$. Holant$(G_1)$ = Holant(G).

It is worth noting that the non-zero multiple factor to a function can not be ignored when considering the value of an individual instance.

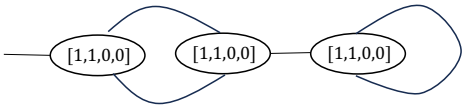
We replace each $[4,2,1,1]$ by the $\{[1,1,0,0]\}$-gate showed in Figure 3, replace each $\sqrt[3]{4}[-1,1]$ by $[1,-1]$, and replace each $(\sqrt[3]{4})^{-1}[-1,2,0]$ by $[1,-2,0]$. The new graph, denoted by G_2, is an instance of planar Holant$(\{[1,1,0,0],[1,-1],[1,-2,0]\})$. For some fixed integers c, d, Holant$(G_1) = (-1)^c(\sqrt[3]{4})^d \cdot$ Holant(G_2). We further replace each occurrence of $[1,-2,0]$ by the gadget showed in Figure 8-(a). This defines G_3 with Holant(G_2) = Holant(G_3). G_3 is an instance of planar Holant$(\{[1,1,0,0],[1,-1],[3,-2]\})$.



(a) Realizing $[1,-2,0]$.



(b) Realizing $[3,1]$.



(c) Realizing $[4,2]$.

Figure 8 The constructions of some gadgets.

We replace each occurrence of $[1,-1]$ and each occurrence of $[3,-2]$ by the gadget with signature $[4,2]$ and the gadget with signature $[3,1]$, respectively. Such gadgets are showed in Figure 8. This defines a new instance G'' of planar Holant$([1,1,0,0])$. Because $1 \equiv 4 \mod 3$, $2 \equiv -1 \mod 3$ and $1 \equiv -2 \mod 3$, so Holant(G'') = Holant$(G_3) \mod 3$.

Since $(-1)^c(\sqrt[3]{4})^d \cdot$Holant$(G_2)$ = Holant(G) is either 0 or 1, Holant$(G) = 0$ if $((-1)^c(\sqrt[3]{4})^d \cdot$ Holant$(G_3) \mod 3) = 0$, and Holant$(G) = 1$ if $((-1)^c(\sqrt[3]{4})^d \cdot$ Holant$(G_3) \mod 3) \neq 0$.

If Holant(G'') can be computed in $O(2^{\varepsilon(\sqrt{cM^2})})$ time for any $\varepsilon > 0$, then #SAT(ϕ) can be solved in $O(2^{\varepsilon' M})$ time for any constant $\varepsilon' > 0$. It is a contradiction to Lemma 8.

▶ **Theorem 21.** *If rETH holds, then there exists some constant $\varepsilon > 0$ such that planar* Holant$([1,1,0,0])$*, i.e., counting all matchings of any given planar 3-regular graph, has no $O(2^{\varepsilon\sqrt{N}})$ time randomized algorithm. N denotes the number of vertices in the input graph.*

6 Fine-grained dichotomy for symmetric Holant*

We develop a fine-grained dichotomy theorem for a class of counting problems: symmetric Holant*, from #3R-VC and #3R-Matching. A function is *degenerate* if it can be written as the tensor power of some unary functions. A Holant problem defined by a set of degenerate functions is trivially computed in polynomial time, so we only consider non-degenerate functions.

▶ **Theorem 22.** *Let $\mathcal{F}$ be a set of non-degenerate symmetric Boolean functions.* Holant$^*(\mathcal{F})$ *is computable in polynomial time if $\mathcal{F}$ satisfies one of the following cases.*

1. *Every function in $\mathcal{F}$ is of arity no more than two.*
2. *There exist two constants a and b, which are not both zero and depending only on $\mathcal{F}$, such that for all functions $[x_0, x_1, ..., x_n] \in \mathcal{F}$ one of the two conditions is satisfied: (1) for every $k = 0, 1, ..., n-2$, there is $ax_k + bx_{k+1} - ax_{k+2} = 0$; (2) $n = 2$ and the function $[x_0, x_1, x_2]$ is of the form $[2a\lambda, b\lambda, -2a\lambda]$ for some constant $\lambda \in \mathbb{C}$.*
3. *For every function $[x_0, x_1, ..., x_n] \in \mathcal{F}$ one of the two conditions is satisfied: (1) for every $k = 0, 1, ..., n-2$, there is $x_k + x_{k+2} = 0$; (2) $n = 2$ and the function $[x_0, x_1, x_2]$ is of the form $[\lambda, 0, \lambda]$ for some constant $\lambda \in \mathbb{C}$.*

Otherwise, there exists some constant $\varepsilon > 0$ such that it has no $2^{\varepsilon N}$ time deterministic algorithm under #ETH. N denotes the number of vertices in the input graph.

For planar Holant$^*(\mathcal{F})$ *which does not satisfy the tractable conditions, there also exists some constant $\varepsilon' > 0$ such that it has no $2^{\varepsilon'\sqrt{N}}$ time deterministic algorithm under #ETH. Besides, it has no $2^{\varepsilon'\sqrt{N}}$ time randomized algorithm under rETH.*

Proof. We follow the original proofs in [4] to develop a fine-grained dichotomy theorem for the class symmetric Holant*. Given the problem Holant$^*(\mathcal{F})$, the problem has a polynomial time algorithm [4] if $\mathcal{F}$ satisfies the tractable conditions; otherwise, Cai et al. used gadget constructions and holographic transformations to build the polynomial reductions from Holant$^*(=_3 | OR_2)$ or Holant$^*([1, 0, 0, 1])$ to Holant$^*(\mathcal{F})$. Since gadget constructions and holographic transformations all preserve the $2^{\Omega(N)}$ or $2^{\Omega(\sqrt{N})}$ time lower bound. Besides, these reductions only use planar gadgets.

So the problem Holant$^*(\mathcal{F})$, with $\mathcal{F}$ violating the tractable conditions, has the $2^{\Omega(N)}$ time lower bound under #ETH. Moreover, it has the $2^{\Omega(\sqrt{N})}$ time lower bound if the inputs are restricted to planar graphs. ◀

7 Conclusion

Based on #ETH, we prove the tight $2^{\Omega(N)}$ time lower bounds for Holant$(=_3 | OR_2)$ and Holant$([1, 0, 0, 1])$ by Theorem 11 and Theorem 13. And we prove the tight $2^{\Omega(\sqrt{N})}$ time lower bounds for pl-Holant$^*(=_3 | OR_2)$ and pl-Holant$^*([1, 0, 0, 1])$ under #ETH. We also present a fine-grained dichotomy theorem for a class of counting problems, symmetric Holant*.

One of the further works is the development of the fine-grained dichotomy theorem under #ETH. The development is challenged when the inputs of counting problems are restricted to planar graphs since we only prove the nearly tight $2^{\Omega(\sqrt{\frac{N}{\log N}})}$ time lower bound for pl-Holant$(=_3 | OR_2)$ and pl-Holant$([1, 0, 0, 1])$. The problem that whether pl-Holant$(=_3 | OR_2)$ or pl-Holant$([1, 0, 0, 1])$ has $2^{o(\sqrt{N})}$ time algorithm or not, under #ETH, is still open. However, this paper still presents a novelty application of polynomial interpolation, which can be popularized to prove the nearly tight $2^{\Omega(\sqrt{\frac{N}{\log N}})}$ time lower bound for more generalized planar counting problems.

Based on rETH, we prove the tight $2^{\Omega(\sqrt{N})}$ time lower bound for #pl-3R-VC and #pl-3R-Matching. The reductions under rETH need to pay close attention to the exact values of the generated instances, so their generality is limited. However, these reductions still provide a method that avoids interpolation for developing the tight lower bound for planar counting problems.

References

1 Cornelius Brand, Holger Dell, and Marc Roth. Fine-grained dichotomies for the tutte plane and boolean #csp. *Algorithmica*, 81(2):541–556, 2019.

2 Jin-Yi Cai, Sangxia Huang, and Pinyan Lu. From holant to #csp and back: Dichotomy for holantc problems. *Algorithmica*, 64(3):511–533, November 2012. doi:10.1007/s00453-012-9626-6.

3 Jin-Yi Cai, Pinyan Lu, and Mingji Xia. Holographic algorithms by fibonacci gates and holographic reductions for hardness. In *Proceedings of the 2008 49th Annual IEEE Symposium on Foundations of Computer Science*, FOCS '08, pages 644–653, USA, 2008. IEEE Computer Society. doi:10.1109/FOCS.2008.34.

4 Jin-Yi Cai, Pinyan Lu, and Mingji Xia. Holant problems and counting csp. In *Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing*, STOC '09, pages 715–724, New York, NY, USA, 2009. Association for Computing Machinery. doi:10.1145/1536414.1536511.

5 Chris Calabro, Russell Impagliazzo, Valentine Kabanets, and Ramamohan Paturi. The complexity of unique k-sat: An isolation lemma for k-cnfs. *Journal of Computer and System Sciences*, 74(3):386–393, 2008. Computational Complexity 2003. doi:10.1016/j.jcss.2007.06.015.

6 Katrin Casel, Henning Fernau, Mehdi Ghadikoalei, Jérôme Monnot, and Florian Sikora. Extension of vertex cover and independent set in some classes of graphs. *International Conference on Algorithms and Complexity (ICAC 2019)*, 11485:124–136, April 2019. doi:10.1007/978-3-030-17402-6_11.

7 Radu Curticapean. Block interpolation: A framework for tight exponential-time counting complexity. *Information and Computation*, 261:265–280, 2018.

8 Holger Dell, Thore Husfeldt, Dániel Marx, Nina Taslaman, and Martin Wahlen. Exponential time complexity of the permanent and the tutte polynomial. *ACM Transaction on Algorithms*, 10(4):21:1–21:32, 2014.

9 J. Flum and M. Grohe. The parameterized complexity of counting problems. In *The 43rd Annual IEEE Symposium on Foundations of Computer Science, 2002. Proceedings.*, pages 538–547, 2002. doi:10.1109/SFCS.2002.1181978.

10 Mark K. Goldberg, Thomas H. Spencer, and Dave A. Berque. A low-exponential algorithm for counting vertex covers. *Journal of Graph Theory*, 1992.

11 Catherine Greenhill. The complexity of counting colourings and independent sets in sparse graphs and hypergraphs. *Comput. Complex.*, 9(1):52–72, January 2000. doi:10.1007/PL00001601.

12 Russell Impagliazzo and Ramamohan Paturi. On the complexity of k-sat. *Journal of Computer and System Sciences*, 62(2):367–375, 2001.

13 Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. Which problems have strongly exponential complexity? *Journal of Computer and System Sciences*, 63(4):512–530, 2001.

14 Michael Kowalczyk and Jin-Yi Cai. Holant problems for 3-regular graphs with complex edge functions. *Theory of Computing Systems*, 59(1):133–158, July 2016. doi:10.1007/s00224-016-9671-7.

15 Ying Liu. Exponential time complexity of the complex weighted boolean #csp. In Weili Wu and Guangmo Tong, editors, *Computing and Combinatorics*, pages 83–96, Cham, 2024. Springer Nature Switzerland.

16 Dániel Marx, Govind S. Sankar, and Philipp Schepper. Degrees and Gaps: Tight Complexity Results of General Factor Problems Parameterized by Treewidth and Cutwidth. In Nikhil Bansal, Emanuela Merelli, and James Worrell, editors, *48th International Colloquium on Automata, Languages, and Programming (ICALP 2021)*, volume 198 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 95:1–95:20, Dagstuhl, Germany, 2021. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ICALP.2021.95.

17 Dimitrios M. Thilikos. Compactors for parameterized counting problems. *Computer Science Review*, 39:100344, 2021. doi:10.1016/j.cosrev.2020.100344.

18 Salil P. Vadhan. The complexity of counting in sparse, regular, and planar graphs. *SIAM Journal on Computing*, 31(2):398–427, 2001. doi:10.1137/S0097539797321602.

19 Leslie G Valiant. The complexity of computing the permanent. *Theoretical computer science*, 8(2):189–201, 1979.

20 Leslie G. Valiant. Accidental algorthims. In *Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science*, FOCS '06, pages 509–517, USA, 2006. IEEE Computer Society. doi:10.1109/FOCS.2006.7.

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

22 Yitong Yin and Chihao Zhang. Approximate counting via correlation decay on planar graphs. In *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '13, pages 47–66, USA, 2013. Society for Industrial and Applied Mathematics.

23 Liu Ying. The complexity of contracting planar tensor network. *ArXiv*, abs/2001.10204, 2020.