Parallel Sampling via Counting

Nima Anari[1], Ruiquan Gao[1], and Aviad Rubinstein[1]

[1]Stanford University, {anari,ruiquan,aviad}@stanford.edu

Abstract

We show how to use parallelization to speed up sampling from an arbitrary distribution μ on a product space $[q]^n$, given oracle access to counting queries: $\mathbb{P}_{X\sim\mu}[X_S = \sigma_S]$ for any $S \subseteq [n]$ and $\sigma_S \in [q]^S$. Our algorithm takes $O(n^{2/3} \cdot \mathrm{polylog}(n, q))$ parallel time, to the best of our knowledge, the first sublinear in n runtime for arbitrary distributions. Our results have implications for sampling in autoregressive models. Our algorithm directly works with an equivalent oracle that answers conditional marginal queries $\mathbb{P}_{X\sim\mu}[X_i = \sigma_i \mid X_S = \sigma_S]$, whose role is played by a trained neural network in autoregressive models. This suggests a roughly $n^{1/3}$-factor speedup is possible for sampling in any-order autoregressive models. We complement our positive result by showing a lower bound of $\widetilde{\Omega}(n^{1/3})$ for the runtime of any parallel sampling algorithm making at most $\mathrm{poly}(n)$ queries to the counting oracle, even for $q = 2$.

1 Introduction

The seminal work of Jerrum, Valiant, and Vazirani [JVV86] established an algorithmic equivalence between the tasks of *approximate sampling* and *approximate counting*, for the ubiquitous class of self-reducible problems. This key equivalence is at the heart of the Monte Carlo Markov Chain approach to approximate counting [ŠVV09], which has enabled breakthroughs like approximating the permanent [JSV04] or the volume of convex sets [DFK91]. In this paper, we focus on one side of this equivalence, sampling via counting.

Self-reducibility, in its most widely applied form, concerns distributions μ on a product space $[q]^n$ and their pinnings: conditional distributions obtained by selecting a subset $S \subseteq [n]$ and partial configuration $\sigma_S \in [q]^S$ and conditioning $X \sim \mu$ to have coordinates in S pinned to σ_S: $X_S = \sigma_S$. In this setting, sampling means producing a random X distributed according to a specified pinning of μ. Counting, on the other hand, refers to computing the partition functions of pinnings: $\mathbb{P}_{X\sim\mu}[X_S = \sigma_S]$.[1] Sampling via counting is in fact very easy to describe in this setting. Assuming access to a counting oracle, we can produce samples from μ via the following *autoregressive generation process*:

Initialize $\sigma \leftarrow \emptyset$
for $i = 1, \dots, n$ **do**
  for $x \in [q]$ **do**
    $p_x \leftarrow \mathbb{P}_{X\sim\mu}\left[X_i = x \mid X_{[i-1]} = \sigma_{[i-1]}\right]$
  $\sigma_i \leftarrow$random sample in $[q]$ distributed $\sim (p_1, \dots, p_q)$
return σ

Note that we only need to use the counting oracle to compute the computationally equivalent *conditional marginals*:

$$\mathbb{P}_{X\sim\mu}\left[X_i = x \mid X_{[i-1]} = \sigma_{[i-1]}\right] = \frac{\mathbb{P}_{X\sim\mu}\left[X_{[i-1]} = \sigma_{[i-1]}, X_i = x\right]}{\mathbb{P}_{X\sim\mu}\left[X_{[i-1]} = \sigma_{[i-1]}\right]}.$$

This process, despite its simplicity, is how the widely successful autoregressive models generate their output [see, e.g., LM11; VKK16; Vas+17; Dev+18; Yan+19; Bro+20]. State-of-the-art large language models, or even very competitive vision models, train a neural network to answer *conditional marginal queries* and then use the aforementioned process to generate samples. In the context of language models, $[q]$ represents a token space, and n is the length of generated text or context length, while in pixel-space vision models, $[q]$ is possible values for a pixel, and n is the number of pixels in the image.

One downside of this simple sampling process is that it is extremely sequential. One has to generate coordinates $1, \dots, i-1$, to know which conditional marginals need to be queried in the ith iteration. So, it is natural to ask if there is a more *parallelizable* sampling process. More precisely, suppose that an oracle[2] can answer conditional marginal queries of the form $\mathbb{P}[X_i = x \mid X_S = \sigma_S]$, and we can interact with this oracle in rounds, each time asking polynomially many queries simultaneously. We are interested in finding the *adaptive complexity* of sampling:

Question 1. *What is the minimum number of rounds before we can produce a sample?*

[1] In the literature, often μ is assumed to be an unnormalized measure. The partition function for unnormalized measures is simply $\mu(\{X \in [q]^n \mid X_S = \sigma_S\})$. Counting algorithms for unnormalized measures and normalized measures are easily reducible to each other, so w.l.o.g. we assume μ is normalized.

[2] E.g., a neural network in learned autoregressive models.

At first glance, it might seem that $\simeq n$ is roughly the optimal number of rounds. Indeed, if we are restricted to asking queries where we always pin a prefix $X_{[i-1]}$ and ask for the conditional marginal of the next coordinate X_i, not much better is possible. Imagine the adversarially chosen distribution μ being a Dirac delta on a single randomly chosen $\sigma \in [q]^n$. One cannot "guess" more than $\widetilde{O}(1)$ coordinates of σ at a time, and thus any query pinning more than $\widetilde{O}(1)$ new coordinates is useless. Thus it takes $\widetilde{\Omega}(n)$ rounds to find the hidden σ.

Perhaps surprisingly, we show that when pinning is allowed on *any subset* of the coordinates, we can significantly improve over n. For details of the algorithm, see Section 3.

Theorem 2 (Main)**.** *There is an algorithm that produces a random sample from any distribution μ on $[q]^n$ by interacting in rounds with an oracle that answers conditional marginal queries, with each query returning $\mathbb{P}_{X\sim\mu}[X_i = x \mid X_S = \sigma_S]$ for all $x \in [q]$. The total number of queries is $O(n)$, and the expected number of rounds is*

$$O(n^{2/3} \cdot \min\{\log^{2/3} n \cdot \log q, q^{1/3} \log^{1/3} q\}).$$

We note that, although we mostly care about parallelizing interactions with the oracle, our algorithm's internal computation can also be parallelized, and up to polylogarithmic factors, the runtime on a PRAM would be the same as the bound in Theorem 2. We also note that the guarantee on the expected number of rounds for our algorithm also holds with high probability at the cost of extra logarithmic factors, see Theorem 26.

Remark 3. In autoregressive models, especially large language models, q is usually very large, but Theorem 2 has a mild dependency on q, at most logarithmic. Since autoregressive models are often run on hardware already capable of massive amounts of parallelism, e.g., GPUs or TPUs, one can expect our algorithm to speed up generation time even in practice. We leave experimental evaluation to future works, but we also note two potential issues. First, while many autoregressive models, such as XLNet [Yan+19] or generally any-order autoregressive models [SSE22], allow pinning of any subset, many others only allow pinning of prefixes; as noted before, no significant parallel speedup is possible for just prefix pinnings. Second, in practice, the oracle's role is played by a trained neural network, which clearly returns only approximate answers. While we can handle approximate oracles, see Section 3.4, the guarantees we need in theory might not hold in practice.

One might wonder if the number of rounds can be further improved, perhaps by using a different algorithm. In a dream scenario, would a polylogarithmic number of rounds be feasible? We answer this question *negatively*, by providing a lower bound of $\widetilde{\Omega}(n^{1/3})$ for any algorithm.

Theorem 4 (Lower bound, informal)**.** *Even for $q = 2$, any algorithm sampling from arbitrary distributions on $[q]^n$ needs to interact with the conditional marginal oracle for at least $\widetilde{\Omega}(n^{1/3})$ rounds.*

For the more formal statement of our lower bound, see Section 5. This shows that the optimal number of rounds, while sublinear in n, must still be a polynomially large function of n, at least with no further assumption on the distribution μ.

1.1 Related Work

Interest in parallel sampling started decades ago. As an early example, Mulmuley, Vazirani, and Vazirani [MVV87], having found an algorithm to generate perfect matchings in parallel, asked if a *uniformly random* one can also be generated in parallel. Teng [Ten95] provided negative evidence for this. Recently, there has been a significantly increased interest in parallel sampling algorithms.

Markov chains, arguably the most successful sampling tool, are naïvely sequential, but recent works have shown techniques for parallelizing some classes of Markov chains, including Glauber dynamics, under tractability conditions on the distribution μ [FHY21; LY22; Lee23]. We note that even sequential implementations of Markov chains such as Glauber dynamics take exponential time on *worst-case distributions* μ, so it is natural that these works need further assumptions on μ.

Most related to our work, parallel sampling was raised as an open question by Anari, Hu, Saberi, and Schild [Ana+20] for several challenge distributions that admit parallel (NC, i.e., polylogarithmic time on polynomially many machines) counting algorithms. These include the distributions of uniformly random arborescences, directed Eulerian tours, planar perfect matchings, and determinantal point processes. They showed polylogarithmic sampling is possible for one of these challenges: sampling uniformly random arborescences. Later, Anari, Burgess, Tian, and Vuong [Ana+23a] showed polynomial parallel speedups are possible for the class of *entropically independent* distributions, which included all challenges except for *planar perfect matchings*. Most recently, Anari, Huang, Liu, Vuong, Xu, and Yu [Ana+23b] achieved polylogarithmic sampling for all challenge distributions except for *planar perfect matchings*, using the stronger "weighted counting oracle." This stronger oracle returns marginals not just after pinnings, but under all "exponential tilts," and interestingly, is what another class of generative AI models, namely *diffusion models*, attempt to learn.

We note that all of these prior works use some tractability assumption about the distribution μ. In fact, none of them are able to nontrivially speed up sampling of *planar perfect matchings*, one of the original challenges. In contrast, in our work, the emphasis is on *arbitrary distributions* μ, as none of the tractability assumptions of prior work is likely to hold for example by distributions learned by autoregressive models. As an application of our results, we show how to nontrivially speed up parallel sampling of *planar perfect matchings* in Section 4.

Recently, generative modeling in AI has produced amazing results. State-of-the-art models, depending on the domain or modality, are often autoregressive or diffusion-based. Given their huge importance in practice, significant attention has been paid to improving the sampling efficiency of these models, particularly via parallelism. For example, Picard iterations in diffusion models [Shi+23] and speculative decoding in autoregressive models [Che+23; LKM23] have shown practical accelerations. There are many other techniques introduced in the literature, evaluated experimentally, by way of example "prediction and forecasting" [WH20] and fixed-point iterations based on Jacobi and Gauss-Seidel equations [Son+21]. To the best of our knowledge, these works focus on real-world distributions and do not theoretically prove an unconditional asymptotic parallel speedup. Interestingly, some of these practical parallelization techniques, for example, speculative decoding, share similarities with our sampling algorithm, Algorithm 2. In speculative decoding, a *draft model*, a much faster but less accurate model, is used to generate guesses sequentially for future tokens and these guesses are "verified" using a larger but more accurate model in parallel. Our algorithm is also based on a guessing and verification paradigm but differs from speculative decoding because we cannot afford to sequentially run a draft model. We emphasize that our work is focused on theoretical guarantees, and works with a single oracle, not tiers of oracles with cost/accuracy tradeoffs.

Finally, adaptive complexity has been studied for many other computational problems, for example, submodular maximization [BRS19; LLV20] and minimization [BS20; CCK21; Cha+22]. Most notably, the parallel complexity of search via a decision oracle was studied in the seminal work of Karp, Upfal, and Wigderson [KUW88], who showed, similarly to our results, that a polynomial speedup, and no better than a polynomial speedup, was possible.

1.2 Techniques

Our algorithm works by modifying the autoregressive sampling process in two ways. First, we choose the order of coordinates according to a uniformly random permutation. Second, to break sequentiality, we generate "guesses" of future coordinates, by computing the marginal of each X_i conditioned on the current pinning, *in parallel*, and sampling from these marginal distributions independently for each i. While these independent samples clearly ignore dependencies between coordinates, we can in a second stage *verify in parallel* that each X_i would have been the sample produced if we had continued sequentially. We advance up to the point where our guesses successfully pass verification and then iterate.

The key idea behind our analysis is that as we pin more and more random coordinates, the dependencies between the remaining coordinates weaken in an *average sense*. This intuitive idea is formalized by the so-called pinning lemmas [RT12; Mon08] which we use in our analysis, see Section 2.1. Weakened dependencies intuitively mean that our guesses are not likely to deviate from what sequential sampling would have produced. This is formally proved in Section 3.

Finally, a tool we use from existing literature on parallel sampling is a universal coupler [LY22]. This is used to ensure consistency between the guessing and verification stages. In both of these phases, for each X_i, we would like to sample from a marginal distribution. Universal couplers ensure that when the marginal distributions are "close", the samples are likely to be exactly equal. We extend the analysis of universal coupling to multiple distributions, as needed by our work, see Section 2.2.

To prove our lower bound, we construct a challenge distribution that is a uniform distribution on an affine subspace of $\mathbb{F}_2^n = \{0,1\}^n$, and we show that it is hard to even output *anything* in its support in fewer than $\widetilde{\Omega}(n^{1/3})$ rounds. We group the coordinates in $[n]$ into roughly $\widetilde{\Omega}(n^{1/3})$ randomly chosen buckets and put varying numbers of affine constraints on each bucket. We prove that with high probability the buckets can only be discovered one at a time, from the most constrained bucket to the least. This is because queries pinning too many coordinates will not be useful at all, as they will violate the constraints of the most constrained bucket. On the other hand, if the number of pinnings is just right for the most-constrained undiscovered bucket, no information is gained about less-constrained buckets; with high probability all of the marginals in the less-constrained buckets remain uniform.

1.3 Organization

In Section 2 we discuss and further develop two of the main tools we use for parallelization: pinning lemmas and universal coupling. In Section 3, we describe our parallel sampling algorithm and prove our main result Theorem 2. In Section 4, we provide an application of Theorem 2 to the problem of sampling planar perfect matchings. In Section 5, we prove a lower bound against all algorithms, i.e., Theorem 4. In Section 6, we prove that our analysis of the algorithm presented in Section 3 is tight, and $n^{2/3}$ cannot be improved for this particular algorithm.

2 Preliminaries

We use $[n]$ to denote the set $\{1, 2, \cdots, n\}$. For any vector x and set S, we use x_S to denote x restricted to S. We use $\pm x$ to denote the interval $[-x, x]$. We use $\mathcal{S}_n$ to denote the set of permutations on n elements. For any two sets A, B, we use $A \times B$ to denote the Cartesian product of A and B, i.e.,

$A \times B = \{(a, b) \mid a \in A, b \in B\}$.

For a distribution μ, we use $x \sim \mu$ to denote that x is sampled from μ. Similarly, for a set S, we use $x \sim S$ to indicate that x is sampled uniformly at random from S.

2.1 Pinning Lemmas

Lemma 5 (pinning lemma [RT12; Mon08])**.** *Let $X_1, X_2, \ldots, X_n$ be random variables, each supported on $\{0,1\}$. For any $\ell \in [n]$, there exists a set S such that $|S| \leq \ell$ and*

$$\mathbb{E}_{X_S}\left[\mathbb{E}_{u,v\sim\binom{[n]}{2}}\left[\mathrm{Cov}(X_u, X_v \mid X_S)^2\right]\right] \leq \frac{O(1)}{\ell}.$$

Definition 6 (entropy)**.** Let X, Y be random variables on $[q]$. The entropy of random variable X is defined to be

$$\mathcal{H}(X) = -\sum_{i\in[q]} \mathbb{P}[X = i] \cdot \log \mathbb{P}[X = i]\,.$$

The conditional entropy of X conditioned on Y is defined to be

$$\mathcal{H}(X \mid Y) = \sum_{i\in[q]} \mathcal{H}(X \mid Y = i) \cdot \mathbb{P}[Y = i].$$

Definition 7 (KL-divergence)**.** For a pair of distributions ν, μ, we let

$$\mathcal{D}_{\mathrm{KL}}(\nu \parallel \mu) = \mathbb{E}_{x\sim\nu}\left[\log \frac{\nu(x)}{\mu(x)}\right].$$

Abusing notation, we extend the definition to random variables. If $X \sim \nu, Y \sim \mu$, we use $\mathcal{D}_{\mathrm{KL}}(X \parallel Y)$ to denote $\mathcal{D}_{\mathrm{KL}}(\nu \parallel \mu)$.

Lemma 8. *For any two random variables X, Y,*

$$\mathbb{E}_Y[\mathcal{D}_{\mathrm{KL}}((X \mid Y) \parallel X)] = \mathcal{H}(X) - \mathcal{H}(X \mid Y).$$

Lemma 9 (Pinsker's inequality)**.** *For any two random variables X, Y,*

$$d_{\mathrm{TV}}(X, Y) \leq \sqrt{\frac{1}{2}\,\mathcal{D}_{\mathrm{KL}}(X \parallel Y)}$$

We prove and use the following variant of pinning lemmas.

Lemma 10. *For any integer $\theta > 0$ and any collection of random variables $X = (X_1, \ldots, X_n)$ with support $[q]$,*

$$\mathbb{E}_{X,\sigma\sim\mathcal{S}_n}\left[\sum_{i=\theta}^{n} d_{\mathrm{TV}}\left(X_{\sigma(i)} \;\middle|\; \{X_{\sigma(j)}\}_{j\in[i-\theta]}, X_{\sigma(i)} \;\middle|\; \{X_{\sigma(j)}\}_{j\in[i-1]}\right)^2\right] \leq \frac{(\theta-1)\log q}{2}.$$

Proof. For any permutation $\sigma \in \mathcal{S}_n$ and any $i \in \{\theta, \theta+1, \cdots, n\}$,

$$\mathbb{E}_X\left[d_{\mathrm{TV}}\left(X_{\sigma(i)} \mid \{X_{\sigma(j)}\}_{j\in[i-\theta]}, X_{\sigma(i)} \mid \{X_{\sigma(j)}\}_{j\in[i-1]}\right)^2\right] \le \frac{\mathbb{E}_X\left[\mathcal{D}_{\mathrm{KL}}\left(X_{\sigma(i)} \mid \{X_{\sigma(j)}\}_{j\in[i-\theta]} \,\middle\|\, X_{\sigma(i)} \mid \{X_{\sigma(j)}\}_{j\in[i-1]}\right)\right]}{2} \le \frac{\mathcal{H}\left(X_{\sigma(i)} \mid \{X_{\sigma(j)}\}_{j\in[i-\theta]}\right) - \mathcal{H}\left(X_{\sigma(i)} \mid \{X_{\sigma(j)}\}_{j\in[i-1]}\right)}{2} = \sum_{k=i-\theta}^{i-2} \frac{\mathcal{H}\left(X_{\sigma(i)} \mid \{X_{\sigma(j)}\}_{j\in[k]}\right) - \mathcal{H}\left(X_{\sigma(i)} \mid \{X_{\sigma(j)}\}_{j\in[k+1]}\right)}{2}.$$

Summing over all possible i and taking expectation over all permutations, we have

$$\begin{aligned}
&\sum_{i=\theta}^{n} \mathbb{E}\left[d_{\mathrm{TV}}\left(X_{\sigma(i)} \,\middle|\, \{X_{\sigma(j)}\}_{j\in[i-\theta]}, X_{\sigma(i)} \,\middle|\, \{X_{\sigma(j)}\}_{j\in[i-1]}\right)^2\right] \le \\
&\quad \frac{1}{2}\cdot\sum_{i=\theta}^{n}\sum_{k=i-\theta}^{i-2} \mathbb{E}_{\sigma\sim\mathcal{S}_n}\left[\mathcal{H}\left(X_{\sigma(i)} \mid \{X_{\sigma(j)}\}_{j\in[k]}\right)\right] - \mathbb{E}_{\sigma\sim\mathcal{S}_n}\left[\mathcal{H}\left(X_{\sigma(i)} \mid \{X_{\sigma(j)}\}_{j\in[k+1]}\right)\right] \le \\
&\quad \frac{1}{2}\cdot\sum_{i=\theta}^{n}\sum_{k=i-\theta}^{i-2} \mathbb{E}_{\sigma\sim\mathcal{S}_n}\left[\mathcal{H}\left(X_{\sigma(n)} \mid \{X_{\sigma(j)}\}_{j\in[k]}\right)\right] - \mathbb{E}_{\sigma\sim\mathcal{S}_n}\left[\mathcal{H}\left(X_{\sigma(n)} \mid \{X_{\sigma(j)}\}_{j\in[k+1]}\right)\right] \le \\
&\quad \frac{\theta-1}{2}\cdot\sum_{k=0}^{n-2} \mathbb{E}_{\sigma\sim\mathcal{S}_n}\left[\mathcal{H}\left(X_{\sigma(n)} \mid \{X_{\sigma(j)}\}_{j\in[k]}\right)\right] - \mathbb{E}_{\sigma\sim\mathcal{S}_n}\left[\mathcal{H}\left(X_{\sigma(n)} \mid \{X_{\sigma(j)}\}_{j\in[k+1]}\right)\right] = \\
&\quad\quad \frac{\theta-1}{2}\cdot\mathbb{E}_{\sigma\sim\mathcal{S}_n}\left[\mathcal{H}\left(X_{\sigma(n)}\right) - \mathcal{H}\left(X_{\sigma(n)} \mid \{X_{\sigma(j)}\}_{j\in[n-1]}\right)\right] \le \frac{(\theta-1)\log q}{2}. \qquad \square
\end{aligned}$$

2.2 Universal Coupling

For any integer $q > 0$, let Δ_q be the probability simplex on $[q]$, i.e., $\Delta_q = \{\mu \in [0,1]^q \mid \sum_{i=1}^{q} \mu(i) = 1\}$. The universal coupling is defined as follows:

Definition 11 (Universal Coupling, [LY22])**.** A deterministic function $f : \Delta_q \times [0,1] \to [q]$ is a universal coupling on $[q]$ if, when $r \in [0,1]$ is chosen uniformly at random, for any distribution $\mu \in \Delta_q$ and $x \in [q]$,

$$\mathbb{P}_{r\sim[0,1]}[f(\mu, r) = x] = \mu(x).$$

The universal coupler of Liu and Yin [LY22]. Interpret the uniform random r as a sequence of uniform i.i.d. pairs $(x_1, p_1), (x_2, p_2), \cdots \in [q]\times[0,1]$. Given the distribution μ, the algorithm f picks the smallest index $i \ge 1$ such that $p_i \le \mu(x_i)$ and outputs x_i. See Algorithm 1.

Correctness and efficiency. Liu and Yin [LY22] show that Algorithm 1 is a universal coupler and can be implemented with high probability by choosing only a sequence of $L = O(q \log n)$ pairs $(x_1, p_1), \cdots, (x_L, p_L)$. For the sake of completeness, we present their correctness result below:

Algorithm 1: Universal Coupler MinCoupler of [LY22]

Input: Distribution $\mu \in \Delta_q$, randomness $r \in [0,1]$
Output: A sample from μ
Interpret r as uniform i.i.d. pairs $(x_1, p_1), (x_2, p_2), \cdots \in [q] \times [0,1]$, e.g., by

- using bits of r at odd indices for $x_1, x_2, \cdots$, and
- using bits of r at even indices for $p_1, p_2, \cdots$ in a zig-zag order.

$i^* \leftarrow \min\{i \mid p_i \le \mu(x_i)\}$
return x_{i^*}

Lemma 12 ([Lemma 4.3, LY22]). *Suppose MinCoupler is constructed in Algorithm 1 and i^* is the smallest index chosen by MinCoupler. Then, for any distribution $\mu \in \Delta_q$,*

1. *MinCoupler is a universal coupler: $\forall x \in [q]$,*

$$\mathbb{P}[\textit{MinCoupler}(\mu, r) = x] = \mu(x),$$

2. *i^* follows the geometric distribution with success probability $1/q$.*

Performance. Liu and Yin [LY22] show that for any two distributions $\mu, \nu \in \Delta_q$, the samples produced by Algorithm 1 for the two distributions (using a shared random number r) are different with probability at most $\frac{2\, d_{\mathrm{TV}}(\mu,\nu)}{1+d_{\mathrm{TV}}(\mu,\nu)}$, which is also tight in the worst case.

We generalize the coupling performance to a multi-distribution setting in the following lemma.

Lemma 13. *Consider any $m, q > 0$. Suppose $r \in [0,1]$ is uniformly random. For any distributions $\mu_1, \cdots, \mu_m \in \Delta_q$, the probability that there exist $i, j \in [m]$ such that MinCoupler$(\mu_i, r) \neq$ MinCoupler(μ_j, r) is at most*

$$\frac{\sum_{x\in[q]} \max_{i\in[m]} \mu_i(x) - \min_{i\in[m]} \mu_i(x)}{\sum_{x\in[q]} \max_{i\in[m]} \mu_i(x)}.$$

Proof. For each $j \in [m]$, let i_j^* be the smallest index the universal coupler chooses for μ_j, i.e.,

$$i_j^* := \min\left\{i \;\middle|\; p_i \le \mu_j(x_i)\right\}.$$

If all i_j^* are identical, the coupler's outputs on $\mu_1, \ldots, \mu_m$ are the same. Therefore,

$$\mathbb{P}_r\left[\exists i, j \in [m], \textsc{MinCoupler}(\mu_i, r) \neq \textsc{MinCoupler}(\mu_j, r)\right] \le 1 - \mathbb{P}\left[i_1^* = i_2^* = \cdots = i_m^*\right] = \mathbb{P}\left[\min_{j\in[m]} i_j^* \neq \max_{j\in[m]} i_j^*\right].$$

Observe that

$$\min_{j\in[m]} i_j^* = \min\left\{i \;\middle|\; p_i \le \max_{j\in[m]} \mu_j(x_i)\right\} \quad \text{and} \quad \max_{j\in[m]} i_j^* = \min\left\{i \;\middle|\; p_i \le \min_{j\in[m]} \mu_j(x_i)\right\}.$$

We can thus upper bound

$$\mathbb{P}\left[\max_{j\in[m]} i_j^* \neq \min_{j\in[m]} i_j^*\right] = \sum_{i\geq 1}\mathbb{P}\left[\min_{j\in[m]} i_j^* = i\right]\cdot\mathbb{P}\left[p_i > \min_{j\in[m]}\mu_j(x_i)\;\middle|\; \begin{smallmatrix} p_i\leq \max_{j\in[m]}\mu_j(x_i),\\ \forall i'<i,\ p_{i'}>\max_{j\in[m]}\mu_j(x_{i'})\end{smallmatrix}\right] =$$

$$\sum_{i\geq 1}\mathbb{P}\left[\min_{j\in[m]} i_j^* = i\right]\cdot\mathbb{P}\left[p_i > \min_{j\in[m]}\mu_j(x_i)\;\middle|\; p_i\leq \max_{j\in[m]}\mu_j(x_i)\right] =$$

$$\sum_{i\geq 1}\mathbb{P}\left[\min_{j\in[m]} i_j^* = i\right]\cdot\frac{\mathbb{P}\left[\min_{j\in[m]}\mu_j(x_i) < p_i\leq \max_{j\in[m]}\mu_j(x_i)\right]}{\mathbb{P}\left[p_i\leq \max_{j\in[m]}\mu_j(x_i)\right]} =$$

$$\sum_{i\geq 1}\mathbb{P}\left[\min_{j\in[m]} i_j^* = i\right]\cdot\frac{q^{-1}\cdot\sum_{x\in[q]}\max_{j\in[m]}\mu_j(x) - \min_{j\in[m]}\mu_j(x)}{q^{-1}\cdot\sum_{x\in[q]}\max_{j\in[m]}\mu_j(x)} =$$

$$\frac{\sum_{x\in[q]}\max_{j\in[m]}\mu_j(x) - \min_{j\in[m]}\mu_j(x)}{\sum_{x\in[q]}\max_{j\in[m]}\mu_j(x)}. \qquad \square$$

Remark 14. This matches the upper bound given in [LY22] for two variables.

Finally, we present the performance of the universal coupler on randomly constructed distributions $\mu_1,\cdots,\mu_m$ where $\{\mu_i(x)\}_{i\in[m]}$ is a martingale for each $x\in[q]$. For these distributions, we can have a better upper bound for the probability that the universal coupler does not produce the same outcome for these distributions. The upper bound is only related to the expected ℓ_1 or ℓ_2 distances between the first distribution p_1 and the last distribution p_m, with some $O(\log nq)$ or $O(\sqrt{q})$ factor and some small additive terms.

Lemma 15. *Consider any $m, q > 0$ and any randomly constructed distributions $\mu_1,\cdots,\mu_m\in\Delta_q$. Suppose for any $i\in[m]$, $x\in[q]$, $\mu_i(x)$ is a random variable such that $\sum_{x\in[q]}\mu_i(x) = 1$ for any $i\in[m]$. If for each $x\in[q]$, $\{\mu_i(x)\}_{i\in[m]}$ forms a martingale, then for any $n>0$, $\sum_{x\in[q]}\mathbb{E}[\max_{i\in[m]}\mu_i(x) - \min_{i\in[m]}\mu_i(x)]$ is at most*

$$\min\left\{O(\log nq)\cdot\mathbb{E}[d_{\mathrm{TV}}(\mu_1,\mu_m)] + \frac{1}{n}, O(\sqrt{q})\cdot\mathbb{E}\left[\sum_{x\in[q]}(\mu_m(x)-\mu_1(x))^2\right]^{1/2}\right\}$$

The proof of this lemma follows Doob's maximal inequality on ℓ_1 and ℓ_2 norms. For any martinagle $Y_0, Y_1,\cdots,Y_m$, Doob's maximal inequality gives upper bound for the expected maximum differences between any Y_i and Y_0 simply by the expected differences between Y_m and Y_0.

Lemma 16 (Doob's maximal inequality, [see, e.g., RY13])**.** *For any martingales $Y_0, Y_1,\cdots,Y_m$ and any $p\geq 1, C>0$, the complementary cumulative distribution function of $\max_{i\in[m]}|Y_i - Y_0|$ satisfies*

$$\mathbb{P}\left[\max_{i\in[m]}|Y_i - Y_0|\geq C\right]\leq\frac{\mathbb{E}[|Y_m - Y_0|^p]}{C^p},$$

and for any $p>1$,

$$\mathbb{E}\left[\max_{i\in[m]}|Y_i - Y_0|^p\right]\leq\left(\frac{p}{p-1}\right)^p\cdot\mathbb{E}[|Y_m - Y_0|^p].$$

Corollary 17. *For any martingales $Y_0, Y_1, \cdots, Y_m$ on support $[0,1]$ and any $N > 0$,*

$$\mathbb{E}\left[\max_{i\in[m]} |Y_i - Y_0|\right] \le O(\log N) \cdot \mathbb{E}[|Y_m - Y_0|] + \frac{1}{N}.$$

Proof. Taking $p = 1$ in Lemma 16, we have $\mathbb{P}[\max_{i\in[m]} |Y_i - Y_0| \ge C] \le \mathbb{E}[|Y_m - Y_0|]/C$ for any $C > 0$. Therefore, for any $N > 0$,

$$\mathbb{E}\left[\max_{i\in[m]} |Y_i - Y_0|\right] = \int_0^1 \mathbb{P}\left[\max_{i\in[m]} |Y_i - Y_0| \ge C\right] dC \le \frac{1}{N} + \int_{1/N}^1 \mathbb{P}\left[\max_{i\in[m]} |Y_i - Y_0| \ge C\right] dC \le$$
$$\frac{1}{N} + \int_{1/N}^1 \frac{\mathbb{E}[|Y_m - Y_0|]}{C} dC = O(\log N) \cdot \mathbb{E}[|Y_m - Y_0|] + \frac{1}{N}. \quad \square$$

Proof of Lemma 15. Using the linearity of expectation, we get

$$\sum_{x\in[q]} \mathbb{E}\left[\max_{i\in[m]} \mu_i(x) - \min_{i\in[m]} \mu_i(x)\right] = \sum_{x\in[q]} \mathbb{E}\left[\max_{i\in[m]} \mu_i(x) - \mu_1(x)\right] + \mathbb{E}\left[\mu_1(x) - \min_{i\in[m]} \mu_i(x)\right] \le$$
$$2 \sum_{x\in[q]} \mathbb{E}\left[\max_{i\in[m]} |\mu_i(x) - \mu_1(x)|\right] \quad (1)$$

Using Corollary 17, we can obtain the first upper bound for $\sum_{x\in[q]} \mathbb{E}[\max_{i\in[m]} \mu_i(x) - \min_{i\in[m]} \mu_i(x)]$:

$$Eq.\ (1) \le \sum_{x\in[q]} O(\log nq) \cdot \mathbb{E}\left[|\mu_m(x) - \mu_1(x)|\right] + \frac{1}{nq} = O(\log nq) \cdot \mathbb{E}\left[d_{\mathrm{TV}}(\mu_1(x), \mu_m(x))\right] + \frac{1}{n}.$$

Using Cauchy-Schwarz inequality and Doob's maximal inequality for ℓ_2 norm, we can obtain the second upper bound for $\sum_{x\in[q]} \mathbb{E}[\max_{i\in[m]} \mu_i(x) - \min_{i\in[m]} \mu_i(x)]$:

$$Eq.\ (1) = 2\,\mathbb{E}\left[\sum_{x\in[q]} \max_{i\in[m]} (\mu_i(x) - \mu_1(x))\right] \le 2\,\mathbb{E}\left[\left(\sum_{x\in[q]} \max_{i\in[m]} (\mu_i(x) - \mu_1(x))\right)^2\right]^{1/2} \le$$
$$2\,\mathbb{E}\left[q \sum_{x\in[q]} \max_{i\in[m]} (\mu_i(x) - \mu_1(x))^2\right]^{1/2} \le O(\sqrt{q}) \cdot \mathbb{E}\left[\sum_{x\in[q]} (\mu_m(x) - \mu_1(x))^2\right]^{1/2}. \quad \square$$

3 Sublinear Parallel Sampling via Counting Oracles

In this section, we show our main Theorem 2 that we can (approximately) sample from a distribution after a sublinear number of rounds (in terms of the number of variables) of querying a polynomial number of the distribution's counting oracles.

3.1 Algorithm

We present our algorithm in Algorithm 2. Let $u_1, \ldots, u_n$ be the random seeds of the algorithm. The algorithm also shuffles the coordinates with a uniformly random permutation that we ignore in this description for simplicity.

We use the universal coupler MinCoupler constructed in Algorithm 1. We let $x \in [q]^n$ denote a sample from the target distribution generated using the naive sequential algorithm that iteratively samples the i-th entry conditioning on all previous entries:

$$x_i \leftarrow \textsc{MinCoupler}\left(X_i \;\middle|\; \left\{X_j = x_j\right\}_{j\in[i-1]}, u_i\right).$$

The goal of the algorithm is to sample faster than one coordinate per iteration. The algorithm maintains an index a where the a-th and earlier entries are all correctly sampled. At the t-th iteration, the algorithm attempts to re-sample all entries after a by conditioning on the a-th and earlier entries:

$$x_i^t \leftarrow \textsc{MinCoupler}\left(X_i \;\middle|\; \left\{X_j = x_j^{t-1}\right\}_{j\in[a]}, u\right).$$

Then, the algorithm uses x^t to find the earliest entry a' where sampling conditioning on $x^t_{j\in[a'-1]}$ differs from sampling conditioning on $x^{t-1}_{j\in[a]}$ and immediately fixes the entry a': then a' is a new index where the a'-th and earlier entries are all correctly sampled.

Algorithm 2: Parallel sampling on product spaces

Input: Counting oracle μ, universal coupler MinCoupler on $[q]$
Output: A sample in $[q]^n$
Sample a permutation $\sigma \leftarrow \mathrm{uniform}(\mathcal{S}_n)$
Sample i.i.d.s $u_1, u_2, \cdots, u_n \leftarrow \mathrm{uniform}([0,1])$
Initialize $a \leftarrow 0$
Initialize $t \leftarrow 0, x^0 = 1^n$
while true do
  $t \leftarrow t + 1$
  for $i \in [n]$ **in parallel do**
    $y^t_{\sigma(i)} \leftarrow \textsc{MinCoupler}\left(X_{\sigma(i)} \;\middle|\; \left\{X_{\sigma(j)} = x^{t-1}_{\sigma(j)}\right\}_{j\in[a]}, u_i\right)$
  for $i \in [n]$ **in parallel do**
    $x^t_{\sigma(i)} \leftarrow \textsc{MinCoupler}\left(X_{\sigma(i)} \;\middle|\; \left\{X_{\sigma(j)} = y^t_{\sigma(j)}\right\}_{j\in[i-1]}, u_i\right)$
  if $x^t = y^t$ **then**
    return x^t
  $a \leftarrow \min\left\{i \in [n] : y^t_{\sigma(i)} \neq x^t_{\sigma(i)}\right\}$
  if $a = n$ **then**
    return x^t

3.2 Correctness

We consider a function $\tilde{x} : \mathcal{S}_n \times [0,1]^n \to [q]^n$ defined iteratively as follows:

$$\tilde{x}_i(\sigma, u) = \textsc{MinCoupler}\left(X_{\sigma(i)} \;\middle|\; \left\{X_{\sigma(j)} = \tilde{x}_j(\sigma, u)\right\}_{j\in[i-1]}, u_i\right). \tag{2}$$

For each $i \in [n]$, because MinCoupler is a universal coupler, $\tilde{x}_i(\sigma, u)$ follows the marginal distribution of $X_{\sigma(i)}$ conditioning on $X_{\sigma(1)} = \tilde{x}_1(\sigma, u), X_{\sigma(2)} = \tilde{x}_2(\sigma, u), \cdots, X_{\sigma(i-1)} = \tilde{x}_{i-1}(\sigma, u)$ (considering only the randomness of u_i). Therefore, this function can serve as an objective output of the algorithm when we have fixed the randomness σ and u.

Lemma 18. *If Algorithm 2 always outputs $x^t_{\sigma(i)} = \tilde{x}_i(\sigma, u)$, it samples perfectly from the distribution of X.*

Let a^t be the value of a at the end of round t (if the algorithm does not terminate with $x^t = y^t$ before updating the value of a in round t). For simplicity, we suppose $a^0 = 0$. Next, we show that the vector x^t produced by the algorithm in each round matches this objective vector in the first a^t entries.

Lemma 19. *For any σ, u, after each round t of Algorithm 2,*

$$\forall i \in [a^t], \quad x^t_{\sigma(i)} = \tilde{x}_i(\sigma, u).$$

Proof. According to the definition of a^t, we have $\forall i \in [a^t - 1]$, $x^t_{\sigma(i)} = y^t_{\sigma(i)}$. Therefore, according to the definition of x^t, we have for any $i \in [a^t]$,

$$x^t_{\sigma(i)} = \textsc{MinCoupler}\left(X_{\sigma(i)} \;\middle|\; \left\{X_{\sigma(j)} = x^t_{\sigma(j)}\right\}_{j\in[i-1]},\ u_i\right). \tag{3}$$

Note that this recursion matches the recursion Eq. (2) used in the definition of $\tilde{x}_i(\sigma, u)$ for any $i \in [a^t]$. Therefore, $\forall i \in [a^t]$, $x^t_{\sigma(i)} = \tilde{x}_i(\sigma, u)$. □

To show that Algorithm 2 is making progress every iteration, we prove a^t is (strictly) monotone in terms of t.

Lemma 20. *a^t is strictly increasing with t.*

Proof. According to the definition of y^t, for any $i \in [a^{t-1}]$, $y^t_{\sigma(i)} = x^{t-1}_{\sigma(i)} = \tilde{x}_i(\sigma, u)$. Therefore, according to the definition of x^t, for any $i \in [a^{t-1} + 1]$,

$$\begin{aligned}
x^t_{\sigma(i)} &= \textsc{MinCoupler}\left(X_{\sigma(i)} \;\middle|\; \left\{X_{\sigma(j)} = x^{t-1}_{\sigma(j)}\right\}_{j\in[i-1]}\right) \\
&= \textsc{MinCoupler}\left(X_{\sigma(i)} \;\middle|\; \left\{X_{\sigma(j)} = \tilde{x}_j(\sigma, u)\right\}_{j\in[i-1]}\right) && \text{(definition of } a^{t-1}\text{)} \\
&= \tilde{x}_i(\sigma, u) = y^t_{\sigma(i)}. && \text{(definition of } \tilde{x}_i(\sigma, u)\text{)}
\end{aligned}$$

By the definition of a^t, if $x^t \neq y^t$, we get $a^t > a^{t-1} + 1$. □

Note that the algorithm terminates when either of the following two conditions is satisfied in some round t: $a^t = n$ or $x^t = y^t$. Due to Lemma 20, the algorithm always terminates. If it terminates because of the first condition $a^t = n$, due to Lemma 19, the output of the algorithm matches the objective in Lemma 18. Otherwise, because of the definition of x^t and the fact that $x^t = y^t$, the output satisfies Eq. (3), which matches the recursion Eq. (2) used in the definition of $\tilde{x}_i(\sigma, u)$, and thus matches the objective in Lemma 18. As a conclusion, we obtain the correctness of our algorithm.

Lemma 21. *Algorithm 2 returns a sample $x \sim \mu$.*

3.3 Round Complexity

We establish our sublinear round complexity via two steps. First, we ignore the randomness of σ and u, and establish a worst-case round complexity, which can be linear with some choices of σ, u. Second, we show that the expectation of this round complexity is actually $\widetilde{O}(n^{2/3}\log q)$ with the random choices of σ, u. This bound is established by the performance of the universal coupler on randomly constructed distributions that satisfy the martingale property (Lemma 13 and Lemma 15) and a pinning lemma (Lemma 10).

To use this algorithm to nontrivially speed up sampling of *planar perfect matchings*, we also need a tail bound for the round complexity. Because of Markov's inequality, the expected round complexity implies a simple tail bound – the round complexity is less than $c \cdot n^{2/3}\log q$ with probability $\widetilde{\Omega}(1/c)$. At the end of this subsection, we boost this tail bound to $2^{-\widetilde{\Omega}(c)}$ via the simple observation that running several rounds of the algorithm is equivalent to reinitiating the algorithm with a smaller instance.

Worst-case round complexity. First, we consider the randomness σ, u used in the algorithm as part of the input, and give an upper bound for the round complexity. For each $\sigma \in \mathcal{S}_n, u \in [0,1]^n$ and $i \in [n]$, let $\overline{a}_i(\sigma, u)$ be the maximum a such that Algorithm 2 will not correctly sample $X_{\sigma(i)}$ to $\tilde{x}_i(\sigma, u)$, the value of $X_{\sigma(i)}$ in the final output, under the correct conditioning of $X_{\sigma(1)}, \ldots, X_{\sigma(a)}$. Formally, $\tilde{a}_i(\sigma, u)$ is defined as follows:

$$\overline{a}_i(\sigma,u) = \max\left\{a \geq 0 \;\middle|\; \tilde{x}_i(\sigma,u) \neq \textsc{MinCoupler}\left(X_{\sigma(i)} \;\middle|\; \left\{X_{\sigma(j)} = \tilde{x}_j(\sigma,u)\right\}_{j\in[a]}, u_i\right)\right\},$$

where we define the maximum of an empty set to be 0 for simplicity. Using this definition, we can establish a worst-case round complexity of Algorithm 2.

Lemma 22. *For any integer $\theta \geq 1$ and randomness $\sigma \in \mathcal{S}_n, u \in [0,1]^n$, the round complexity of Algorithm 2 is at most*

$$|\{i \in [n] \mid \overline{a}_i(\sigma,u) \geq i - \theta\}| + 1 + \frac{n}{\theta}.$$

Proof. Recall that we define a^t as the value of a after round t, and we define a^0 as 0. The algorithm has at most one round t without computing a^t, when it terminates with $x^t = y^t$. Suppose that the step size of any round t, where the algorithm computes a^t, is the increment $a^t - a^{t-1}$. Based on the step sizes, we divide the rounds into two classes: **small-progress rounds** that have step sizes $< \theta$, and **large-progress rounds** that have step sizes $\geq \theta$. Note that the number of large-progress rounds is at most n/θ because otherwise a will exceed n in some round. It suffices to upper bound the number of small-progress rounds by $|\{i \in [n] : \overline{a}_i(\sigma,u) \geq i - \theta\}|$ to finish the proof.

Consider any round t such that $a^t - a^{t-1} < \theta$. Due to the definition of the algorithm and Lemma 19, the algorithm finds $y^t_{\sigma(i)} = x^t_{\sigma(i)} = \tilde{x}_i(\sigma,u)$ for any $i < a^t$. The algorithm also finds $x^t_{\sigma(a^t)} \neq y^t_{\sigma(a^t)}$. According to the definition of x^t and y^t and Lemma 19, $y^t_{\sigma(a^t)} =$

$$\textsc{MinCoupler}\left(X_{\sigma(a^t+1)} \;\middle|\; \left\{X_{\sigma(j)} = \tilde{x}_j(\sigma,u)\right\}_{j\in[a^{t-1}]}, u_{a^t}\right),$$

and $x^t_{\sigma(a^t)} =$

$$\textsc{MinCoupler}\left(X_{\sigma(a^t+1)} \;\middle|\; \left\{X_{\sigma(j)} = \tilde{x}_j(\sigma,u)\right\}_{j\in[a^t-1]}, u_{a^t}\right) = \tilde{x}_{a^t}(\sigma,u).$$

This implies $\overline{a}_{a^t}(\sigma, u) \geq a^{t-1} > a^t - \theta$. Therefore, $a^t \in \{i \in [n] \mid \overline{a}_i(\sigma, u) \geq i - \theta\}$. Since a^t are strictly increasing (Lemma 20), the number of such small-progress rounds can be upper bounded by $|\{i \in [n] \mid \overline{a}_i(\sigma, u) \geq i - \theta\}|$. □

We note that the cardinality of the set $\{i \in [n] \mid \overline{a}_i(\sigma, u) \geq i-\theta\}$ can be $\Omega(n)$, even under expectation over u. Suppose that $X_1, X_3, \ldots, X_{n-1}$ are sampled independently and uniformly at random, and $X_2 = X_1, X_4 = X_3, \cdots, X_n = X_{n-1}$. For $\theta \geq 2$ and the permutation $\sigma(i) = i$, this set will involve each $2i$ with probability $1/2$ independently.

Remark 23. Because our analysis only needs the **large-progress rounds** to increase a by $\geq \theta$, we could still enjoy the same upper bound if we only resample the first θ entries after a in each iteration. This suggests a more query-efficient implementation Algorithm 3.

Algorithm 3: Query-efficient implementation for each iteration of Algorithm 2

while true do
- $t \leftarrow t + 1$
- $y^t \leftarrow x^{t-1}$
- **for** $i \in \{a + 1, \ldots, \min\{a + \theta, n\}\}$ **in parallel do**
  - $y^t_{\sigma(i)} \leftarrow \textsc{MinCoupler}\left(X_{\sigma(i)} \;\middle|\; \left\{X_{\sigma(j)} = x^{t-1}_{\sigma(j)}\right\}_{j\in[a]}, u_i\right)$
- **for** $i \in \{a + 1, \ldots, \min\{a + \theta, n\}\}$ **in parallel do**
  - $x^t_{\sigma(i)} \leftarrow \textsc{MinCoupler}\left(X_{\sigma(i)} \;\middle|\; \left\{X_{\sigma(j)} = y^t_{\sigma(j)}\right\}_{j\in[i-1]}, u_i\right)$
- $a \leftarrow \min\left\{i \in \{a + 1, \ldots, \min\{a + \theta, n\}\} : y^t_{\sigma(i)} \neq x^t_{\sigma(i)}\right\} \cup \{\min\{a + \theta, n\} + 1\}$
- **if** $a \geq n$ **then**
  - **return** x^t

Expected round complexity. Next, we consider the randomness σ, u and show an improved expected round complexity for the algorithm. Based on the worst-case analysis, we give a better bound for the expected cardinality of $\{i \in [n] \mid \overline{a}_i(\sigma, u) \geq i - \theta\}$ for some sophisticated choice of θ. Due to the linearity of expectation, we can separately upper bound the probability of $\overline{a}_i(\sigma, u) \geq i-\theta$ for each $i \in [n]$. Only considering the randomness of u, we can obtain the following lemma, which upper bounds the probability by the total variation distance between the conditional distributions of $X_{\sigma(i)}$ under fully conditioning of all previous variables ($X_{\sigma(1)}$ to $X_{\sigma(i-1)}$) and partial conditioning of some previous variables ($X_{\sigma(1)}$ to $X_{\sigma(i-\theta)}$).

Lemma 24. *For any $i \geq \theta$ and any permutation $\sigma \in \mathcal{S}_n$, the probability of $\overline{a}_i(\sigma, u) \geq i - \theta$ is at most*

$$O(\min\{\log nq, \sqrt{q}\}) \cdot \mathbb{E}_X\left[d_{\mathrm{TV}}\left(X_{\sigma(i)} \;\middle|\; \{X_{\sigma(j)}\}_{j\in[i-1]}, X_{\sigma(i)} \;\middle|\; \{X_{\sigma(j)}\}_{j\in[i-\theta]}\right)^2\right]^{1/2} + \frac{1}{n},$$

where the probability is taken only over the randomness of $u_1, \cdots, u_n$.

Proof. According to the definition of $\overline{a}_i(\sigma, u)$, $\overline{a}_i(\sigma, u) \geq i - \theta$ when there exists $a \in [i - \theta, i - 1]$ such that $\textsc{MinCoupler}(X_{\sigma(i)} \mid \{X_{\sigma(j)} = \tilde{x}_j(\sigma, u)\}_{j\in[i-1]}, u_i) \neq \textsc{MinCoupler}(X_{\sigma(i)} \mid \{X_{\sigma(j)} = \tilde{x}_j(\sigma, u)\}_{j\in[a]}, u_i)$. Equivalently, when we apply the universal coupler on all variables $X_i|\{X_{\sigma(j)}\}_{j\in[a]}$,

where a can be any integer in $[i-\theta, i-1]$, the coupler produces different outcomes for two of them. Let $\{\mu_a(x)\}_{x\in[q]}$ denote the variables characterizing the randomly constructed distribution of $X_{\sigma(i)}|\{X_{\sigma(j)}\}_{j\in[a]}$, where the randomness is taken over the random conditioning of $\{X_{\sigma(j)}\}_{j\in[a]}$, i.e.,

$$\forall x \in [q],\ \mu_a(x) = \mathbb{P}\left[X_{\sigma(i)} = x \,\middle|\, \{X_{\sigma(j)}\}_{j\in[a]}\right]$$

We have that

$$\mathbb{P}_u[\overline{a}_i(\sigma,u) \geq i-\theta] = \mathbb{P}_u[\exists a, a' \in [i-\theta, i-1], \textsc{MinCoupler}(\mu_a, u_i) \neq \textsc{MinCoupler}(\mu_{a'}, u_i)]\,.$$

Further, note that $\{\mu_a(x)\}_{a\in[i-\theta,i-1]}$ forms a martingale, where randomness is taken over $u_1, \cdots, u_{i-1}$. Because of Lemma 13 and Lemma 15, we can upper bound $\mathbb{P}_u[\overline{a}_i(\sigma,u) \geq i-\theta]$ by the following terms.

$$\mathbb{E}\left[\sum_{x\in[q]} \max_{a\in[i-\theta,i-1]} \mu_a(x) - \min_{a\in[i-\theta,i-1]} \mu_a(x)\right] \leq$$

$$\min\left\{O(\log nq)\cdot \mathbb{E}[d_{\mathrm{TV}}(\mu_{i-\theta}, \mu_{i-1})] + \frac{1}{n}, O(\sqrt{q})\cdot \mathbb{E}\left[\sum_{x\in[q]} (\mu_{i-1}(x) - \mu_{i-\theta}(x))^2\right]^{1/2}\right\} \leq$$

$$O(\min\left\{\log nq, \sqrt{q}\right\})\cdot \mathbb{E}_X\left[d_{\mathrm{TV}}\left(\mu_{i-1}, \mu_{i-\theta}\right)^2\right]^{1/2} + \frac{1}{n}. \quad \square$$

Then, applying the pinning lemma (Lemma 10), we can obtain a better average round complexity for Algorithm 2.

Proof of Theorem 2. We show that the expected number of rounds of Algorithm 2 is

$$O(n^{2/3}\cdot \min\{\log^{2/3} n \log q, q^{1/3}\log^{1/3} q\})$$

Consider $\theta = O(n^{1/2})$. According to Lemma 22, the expected number of rounds that Algorithm 2 needs is at most

$$\frac{n}{\theta} + \theta + 1 + \sum_{i\geq\theta} \mathbb{P}_{\sigma,u}[\overline{a}_i(\sigma,u) \geq i-\theta] \tag{4}$$

By Lemma 24, we can upper bound

$$\sum_{i\geq\theta} \mathbb{P}_{\sigma,u}[\overline{a}_i(\sigma,u) \geq i-\theta] \leq$$

$$1 + O(\min\left\{\log nq, \sqrt{q}\right\})\cdot \underbrace{\sum_{i\geq\theta} \mathbb{E}\left[d_{\mathrm{TV}}\left(X_{\sigma(i)} \,\middle|\, \{X_{\sigma(j)}\}_{j\in[i-1]}, X_{\sigma(i)} \,\middle|\, \{X_{\sigma(j)}\}_{j\in[i-\theta]}\right)^2\right]^{1/2}}_{},$$

where we can further upper bound the underbraced term by

$$\leq O(\sqrt{n})\cdot \mathbb{E}\left[\sum_{i\geq\theta} d_{\mathrm{TV}}\left(X_{\sigma(i)} \,\middle|\, \{X_{\sigma(j)}\}_{j\in[i-1]}, X_{\sigma(i)} \,\middle|\, \{X_{\sigma(j)}\}_{j\in[i-\theta]}\right)\right] \leq$$

$$O(\sqrt{n})\cdot \left(\frac{(\theta-1)\log q}{2}\right)^{1/2} \leq O(\sqrt{n\theta\log q})$$

Therefore, if we take $\theta = \frac{n^{1/3}}{\log^{1/3} q (\min\{\log nq, \sqrt{q}\})^{2/3}}$, the expected round complexity of Algorithm 2 can be upper bounded by

$$\frac{n}{\theta} + O(\sqrt{n\theta \log q} \min\{\log nq, \sqrt{q}\}) \leq O(n^{2/3} \cdot \min\{\log^{2/3} n \log q, q^{1/3} \log^{1/3} q\}).$$

Note that our choice of θ guarantees that $\sum_{i \geq \theta} \mathbb{P}_{\sigma,u}[\overline{a}_i(\sigma, u) \geq i - \theta] = O(n/\theta)$. The expected round complexity can also be upper bounded by $O(n/\theta)$. Using our query-efficient implementation, Algorithm 3, the expected total number of queries we make is $O(n)$. □

Tail bounds for the round complexity. Because of Markov's inequality, we can obtain the following corollary:

Corollary 25. *For any input $X_1, X_2, \cdots, X_n$, with probability at least $1/2$, Algorithm 2 terminates in $\widetilde{O}(n^{2/3} \log q)$ rounds.*

Next, we establish a tail bound for the number of rounds used by the algorithm.

Theorem 26. *There exists a constant $M > 0$ such that for any integer $c \geq 1$, Algorithm 2 terminates in $cM \cdot (n \log n)^{2/3} \log q$ rounds with probability at least $1 - 2^{-c}$.*

Proof. Let M be a constant such that Algorithm 2 terminates in $M \cdot (n \log n)^{2/3} \log q$ with probability at least $1/2$. The existence of such M follows Corollary 25. Let R be the variable that denotes the number of rounds Algorithm 2 uses. Next, we prove by induction that for any integer $c \geq 1$, we have

$$\mathbb{P}\left[R > cM \cdot (n \log n)^{2/3} \log q\right] \leq 2^{-c}.$$

The cases where $n = 1$ or $c = 1$ are trivial.

Suppose that we have proved the theorem for any $n < N$. Consider an instance with $n = N$. For any $i \leq n - 1$, let $\mathcal{A}_i$ be the event that Algorithm 2 does not terminate and has $a = i$ after running it for $M \cdot (n \log n)^{2/3} \log q$ rounds. This definition immediately gives us $\sum_{i \leq n-1} \mathbb{P}[\mathcal{A}_i] \leq \frac{1}{2}$. Because of Lemma 20, after running the algorithm for $M \cdot (n \log n)^{2/3} \log q$ rounds, we have $a \geq M \cdot (n \log n)^{2/3} \log q$. Therefore, $\mathbb{P}[\mathcal{A}_0] = 0$ and for any $c \geq 2$,

$$\mathbb{P}\left[R > cM \cdot (n \log n)^{2/3} \log q\right] = \sum_{i \in [n-1]} \mathbb{P}\left[R > cM \cdot (n \log n)^{2/3} \log q \,\middle|\, \mathcal{A}_i\right] \mathbb{P}[\mathcal{A}_i] \leq \frac{1}{2} \cdot \max_{i \in [n-1]} \mathbb{P}\left[R > cM \cdot (n \log n)^{2/3} \log q \,\middle|\, \mathcal{A}_i\right].$$

Next, we show $\mathbb{P}[R > cM \cdot (n \log n)^{2/3} \log q \mid \mathcal{A}_i] \leq 2^{-c+1}$ for any $i \in [n-1]$ to finish the proof. Note that for any $i \in [n-1]$, whether the event $\mathcal{A}_i$ happens is determined by $\sigma(1), \cdots, \sigma(i)$ and $u_1, \cdots, u_i$. Therefore, $\mathcal{A}_i$ is independent of the random permutation in $[n] \setminus \{\sigma(j)\}_{j \in [i]}$ and the randomness of $u_{i+1}, \cdots, u_n$. Further, if $\mathcal{A}_i$ happens after running the algorithm for $M \cdot (n \log n)^{2/3} \log q$ rounds, the algorithm fixes the values of the first i variables and continues with $a = i$. Therefore, conditioning on any $\sigma(1), \cdots, \sigma(i) \in \binom{[n]}{i}$ and $u_1, \cdots, u_i \in [0, 1]$ that cause $\mathcal{A}_i$ to happen, the remaining iterations of the algorithm are equivalent to those in a fresh run of the algorithm on the remaining variables $\{X_j\}_{j \in [n]} \setminus \{X_{\sigma(j)}\}_{j \in [i]}$ conditioning on $\{X_{\sigma(j)} = \tilde{x}_j(\sigma, u)\}_{j \in [i]}$. According to our induction hypothesis, the number of remaining rounds of the algorithm is (strictly) greater than $(c-1)M \cdot ((n-i) \log(n-i))^{2/3} \log q$ with probability at most 2^{-c+1}. Therefore, $\mathbb{P}[R > cM \cdot (n \log n)^{2/3} \mid \mathcal{A}_i]$ is at most 2^{-c+1}. □

3.4 Sampling via Approximate Counting Oracles

We show that our algorithm can also work with approximate counting oracles. Suppose $\hat{\mu}$ is an oracle such that for any $S \subseteq [n]$ and $y \in [q]^S$, with probability at least $1 - \delta$,

$$\hat{\mu}(S, y) \in (1 \pm \epsilon)\, \mathbb{P}_{X \sim \mu}[X_S = y]. \tag{5}$$

At each round of the algorithm, we shall consider an approximate version of the conditional probability distribution. For any permutation σ, indices $0 \le a < i \le n$ and any $y \in [q]^{\{\sigma(j)\}_{j\in[a]}}$, we consider the following distribution ν for $X_{\sigma(i)} | \{X_{\sigma(j)} = y_{\sigma(j)}\}_{j\in[a]}$: for any $x \in [q]$, let $y'(x) \in [q]^{\{\sigma(j)\}_{j\in[a]} \cup \{\sigma(i)\}}$ be the vector such that $y'_{\sigma(i)}(x) = x$ and $y'_{\sigma(j)}(x) = y_{\sigma(j)}$ for any $j \in [a]$, then we have

$$\forall x \in [q], \quad \nu(x) \propto \hat{\mu}\Big(\{\sigma(j)\}_{j\in[a]} \cup \{\sigma(i)\}, y'(x)\Big).$$

In particular, we use $\nu_{i|a}(\sigma, u)$ to denote this approximate version of the distribution for $X_{\sigma(i)} | \{X_{\sigma(j)} = \tilde{x}_j(\sigma, u)\}_{j\in[a]}$. We say that a pair of randomness (σ, u) is good if for any $0 \le a < i \le n$,

$$\textsc{MinCoupler}\Big(X_{\sigma(i)} \,\Big|\, \{X_{\sigma(j)} = \tilde{x}_j(\sigma, u)\}_{j\in[a]}, u_i\Big) = \textsc{MinCoupler}\big(\nu_{i|a}(\sigma, u), u_i\big)\,.$$

We show the following two key lemmas on good randomness. The first states that the approximate counting oracles do not influence the output of the algorithm as long as the randomness is good. The second upper bounds the probability that the randomness is not good. We defer the proofs to Appendix A.1 and Appendix A.2.

Lemma 27. *If the pair of randomness (σ, u) is good, Algorithm 2 with the approximate counting oracle outputs the same vectors as Algorithm 2 with the exact counting oracle in the same number of rounds.*

Lemma 28. *For any pair of randomness (σ, u), it is good with probability at least $1 - O(n^2\epsilon + n^2 q\delta)$.*

Suppose the parameters of the approximate counting oracle satisfy $\delta, \epsilon = O(n^{-3}q^{-1})$. Putting Theorem 26 together with these two lemmas, we can easily obtain the guarantee: if we terminate Algorithm 2 in $O(n^{2/3}\mathrm{poly}\log(n, q))$ rounds, the output distribution is within a total variation distance $O(n^{-1})$ of μ.

4 Applications

In this section, we show an example application of Theorem 2, to the problem of sampling uniformly random perfect matchings in planar graphs. The famous FKT algorithm allows parallel counting of the number of perfect matchings [see, e.g., Ana+23a]. The previous best parallel runtime for this problem is $\widetilde{O}(n^{1/2})$ for planar graphs of size n [Ana+23a].

Remark 29. Two key techniques for deterministic (approximate) counting, namely the tree recursion/correlation decay method [Wei06] and the polynomial interpolation method [Bar16] can often be trivially parallelized. The former involves solving a recursion on a tree of logarithmic depth, and the latter involves enumerating structures of logarithmic size in a host object (e.g., a graph). As such, our results automatically provide a parallel speedup wherever these methods apply.

Theorem 30. *Let $G = (V, E)$ be a planar graph. There exists an algorithm that samples a uniformly random perfect matching in G with a parallel runtime of $\widetilde{O}(n^{1/3})$ and $\mathrm{poly}(n)$ work.*

Proof. Similar to [Ana+23a], we use the planar separator theorem to find a separator of size $O(\sqrt{n})$, sample the portion of the perfect matching incident to the separator, and then recursively sample the rest of the perfect matching in the now-disjoint halves of the graph, in parallel. Our modification is that, while naïvely sampling the separator edges takes $\widetilde{O}(\sqrt{n})$ time, using Theorem 2, we can speed it up to $\widetilde{O}(n^{1/3})$.

To be more specific, given the input graph $G = (V, E)$, we find a planar separator $S \subseteq V$ of size $O(\sqrt{n})$, such that $G - S$ is composed of two smaller graphs, on vertex sets A, B, each of size $\leq (1 - \Omega(1))n$. This can be done in parallel [GM87].

Next, we consider the distribution μ on E^S, where $\mu(x)$ is proportional to the number of perfect matchings that have edge x_v incident to v for all $v \in S$. Note that many configurations $x \in E^S$ are invalid, for example, those where v is not even an endpoint of x_v, or those with clashing edges for two vertices in S. All of these invalid configurations are assigned a measure of 0 under μ. We claim that there is a parallel (NC) counting oracle for μ. Indeed, given a partial pinning, we can check if it is valid, and if so, remove the edges in the pinning from the graph, and simply count perfect matchings in the resulting subgraph. The number of perfect matchings in planar graphs can be efficiently computed in parallel by the FKT algorithm [see, e.g., Ana+23a].

Now we use Algorithm 2 to sample from μ. Once the sample is produced, we remove all the endpoints of this partial matching from G (in particular, this removes all of S), and now we have two disjoint subgraphs of geometrically smaller size. In parallel, we recurse on each.

Note that the total number of calls to Algorithm 2 is $\leq \mathrm{poly}(n)$. By using the tail bounds for our algorithm, Theorem 26, each call finishes in at most $\widetilde{O}(\sqrt{n}^{2/3}) = \widetilde{O}(n^{1/3})$ time, with probability at least $1 - 1/\mathrm{poly}(n)$. Taking a union bound, and using the fact that recursively the subgraphs shrink geometrically, we get that the overall parallel runtime is $\widetilde{O}(n^{1/3})$ with high probability. □

5 Hardness

In this section, we prove that any algorithm cannot approximately sample within a constant total variation distance of arbitrary distribution μ with $n^{1/3-\Omega(1)}$ round complexity and a polynomial number of queries in each round to the exact counting oracle. More generally, we shall prove the following hardness result on parallel search via counting oracles for $q = 2$.

Theorem 31. *For any constant $\delta \in (0, 1]$, any $c \in (0, n^{1-\delta})$ and any (randomized) algorithm* ALG *making at most n^c queries to the counting oracle in each round, there exists an instance $\mu : \{0,1\}^n \to \{0,1\}$ such that* ALG *can only find a solution x (such that $\mu(x) = 1$) with probability at most 0.01 after (strictly) less than $\frac{1}{4} \cdot (\frac{n}{(c+2)\log n})^{1/3}$ rounds of queries.*

In the rest of this section, we use $H = (S, y)$, where $S \subseteq [n]$ and $y \in [q]^S$, to denote a hypercube by $H = \{x \in [q]^n \mid x_S = y\}$. For the abuse of notation, for any function $\mu : \{0,1\}^k \to \{0,1\}$ and any hypercube H, we define $\mu(H) := \sum_{x \in H} \mu(x)$ as the output of the counting oracle.

The (random) hard instances. We consider deterministic algorithms that make at most n^c queries in each round, where $c < n^{1-\delta}$ for some constant $\delta \in (0, 1]$. We randomly partition the n variables into $r = \frac{1}{4}(\frac{n}{(c+2)\log n})^{1/3}$ equal blocks $S_1, S_2, \cdots, S_r$, each with $m = n/r = 4n^{2/3}((c+2)\log n)^{1/3}$ variables. For each block S_i, we choose $a_i = i \cdot 12n^{1/3}((c+2)\log n)^{2/3}$ and define the set of true strings in this block using a random linear code with constraints $m - a_i$: first we independently

and uniformly choose a matrix $B_i \in \{0,1\}^{(m-a_i)\times m}$ and a vector $v_i \in \{0,1\}^{m-a_i}$ at random for each $j \in [m]$; and then we define the boolean function μ_i as follows:

$$\forall x \in \{0,1\}^{S_i}, \mu_i(x) = \mathbb{1}[B_i x = v_i],$$

where all the operations are under $\mathbb{F}_2$. Then, the true strings of the entire function are defined as those projections in each of the blocks are all true, i.e., the entire function μ is then defined as the product of all μ_i:

$$\forall x \in \{0,1\}^n, \mu(x) = \prod_{i\in[r]} \mu_i(x_{S_i}).$$

For any sub-hypercube H, parameterized by S and y, we define H restricted to S_i as $H_{S_i} := \{x \in \{0,1\}^{S_i} \mid x_{S\cap S_i} = y_{S\cap S_i}\}$. Since $S_1, \cdots, S_r$ is a partition of the n variables, we have $H = H_{S_1} \times H_{S_2} \times \cdots \times H_{S_r}$ according to the definition of H. With this fact, we can obtain the following lemma.

Lemma 32. *For any sub-hypercube H of $\{0,1\}^n$, we have*

$$\mu(H) = \prod_{i\in[r]} \mu_i(H_{S_i}).$$

Proof. According to the definition of the function μ and the definition of the counting oracles,

$$\mu(H) = \sum_{x\in H} \mu(x) = \sum_{x\in H} \prod_{i\in[r]} \mu_i(x_{S_i}) = \sum_{x\in H_{S_1}\times\cdots\times H_{S_r}} \prod_{i\in[r]} \mu_i(x_{S_i}) = \prod_{i\in[r]} \sum_{x\in H_{S_i}} \mu_i(x) = \prod_{i\in[r]} \mu_i(H_{S_i}). \quad \square$$

For any sub-hypercube H parameterized by S and y, we define its codimension $\text{co-dim}(H)$ as $|S|$, i.e., the number of variables whose values are fixed in the sub-hypercube. For any function μ_i, we can show that if the codimension of a query H_{S_i} is $\Omega(\log n)$ greater or less than a_i, the query does not give any information about the randomness of B_i, v_i in the construction with high probability. In addition, the proof only uses the randomness of $B_1, v_1, \cdots, B_r, v_r$.

Lemma 33. *For any sub-hypercube H_{S_i} of $\{0,1\}^{S_i}$, if $\text{co-dim}(H_{S_i}) = d$, for any constant $c_1 > 0$, we have*

- *if $d < a_i - c_1 \log n$, $\mu_i(H_{S_i}) = 2^{a_i - d}$ with probability at least $1 - n^{-c_1}$, and*
- *if $d > a_i + c_1 \log n$, $\mu_i(H_{S_i}) = 0$ with probability at least $1 - n^{-c_1}$,*

where the probability is taken over the randomness of B_i and v_i.

Proof. For any $x \in \{0,1\}^{S_i}$ and any $B_i \in \{0,1\}^{(m-a_i)\times m}$, $\mathbb{P}_{v_i\sim\{0,1\}^{m-a_i}}[B_i x = v_i] = 2^{-(m-a_i)}$. Therefore, we can upper bound the probability of $\mu_i(H_{S_i}) \neq 0$ as follows.

$$\mathbb{P}_{B_i,v_i}[\mu_i(H_{S_i}) \neq 0] = \mathbb{P}_{B_i,v_i}[\exists x \in H_{S_i}, \mu_i(x) = 1] \quad \leq \sum_{x\in H_{S_i}} \mathbb{E}_{B_i,v_i}[B_i x = v_i] = 2^{-(m-a_i)} \cdot |H_{S_i}|$$

Since $\text{co-dim}(H_{S_i}) = d$, $|H_{S_i}| = 2^{m-d}$. We have $\mathbb{P}_{B_i,v_i}[\mu_i(H_{S_i}) \neq 0] \leq 2^{m-d-(m-a_i)} = 2^{a_i-d}$. If $d > a_i + c_1 \log n$, we have $\mu_i(H_{S_i}) \neq 0$ with probability at most n^{-c_1}.

On the other hand, we consider the number of solutions $x \in \{0,1\}^{S_i}$ for $B_i x = v_i$ when $d \leq a_i$. For any sub-hypercube H_{S_i} which is parameterized by $S \subseteq S_i$ and $y \in \{0,1\}^S$ (i.e., $H_{S_i} = \{x \in \{0,1\}^{S_i} \mid x_S = y_S\}$) and has codimension d (i.e., $|S| = d$), we can characterize H_{S_i} by d linear equations:

$\forall j \in S, e_j^T x = y_j$, where e_j denotes the indicator vector having value 1 in the j-th entry and having value 0 in all other entries. Therefore, the set $\{x \in \{0,1\}^{S_i} \mid x \in H_{S_i}, B_i x = v_i\}$ can be characterized by $d + m - a_i$ linear equations: $\forall j \in S, e_j^T x = y_j$ and $B_i x = v_i$.

Lemma 34. *For any $m \le n$, $A \in \{0,1\}^{m \times n}$ and $b \in \{0,1\}^m$, if $\mathrm{rank}(A) = m$, then there are 2^{n-m} solutions $x \in \{0,1\}^n$ for the linear equation $Ax = b$ (under $\mathbb{F}_2$).*

According to the above Lemma 34, if vectors in $\{e_j \mid j \in S\}$ and in rows of B_i are linearly independent under $\mathbb{F}_2$, $\mu_i(H_{S_i}) = |\{x \in \{0,1\}^{S_i} \mid x \in H_{S_i}, B_i x = v_i\}| = 2^{a_i - d}$. It is clear that vectors in $\{e_j \mid j \in S\}$ are linearly independent. Consider we start with $V = \{e_j \mid j \in S\}$ and insert rows in B_i into V one by one. When the vectors in V are linearly independent and we insert one row of B_i into V, V becomes linearly dependent only when the row is a linear combination of the vectors in V. Since there are at most $2^{|V|}$ such linear combinations under $\mathbb{F}_2$, the probability that V remains linearly independent after inserting the row is $1 - 2^{|V|-m}$. After inserting all the $m - a_i$ rows into V, V remains linearly independent with probability

$$\prod_{k=d}^{d+m-a_i-1} 1 - 2^{k-m} \ge 1 - \sum_{k=d}^{d+m-a_i-1} 2^{k-m} \ge 1 - 2^{d-a_i}.$$

Therefore, we have $\mu_i(H_{S_i}) = 2^{a_i-d}$ with probability at least $1-2^{d-a_i}$. In particular, if $d < a_i - c_1 \log n$, we have $\mu_i(H_{S_i}) = 2^{a_i-d}$ with probability at least $1 - n^{-c_1}$. □

On the other hand, the random partition $S_1, S_2, \cdots, S_r$ guarantees that any hypercube has approximately equal codimension in each block with high probability.

Lemma 35. *For any $1 \le k < i \le n$, any $c_2 > 0$, and any sub-hypercube H, the probability that $\text{co-dim}(H_{S_i})$ is in the range $\text{co-dim}(H_{S_k \cup \cdots \cup S_r})/(r-k+1) \pm \sqrt{3c_2 m \log n}$ is at least $1 - 2n^{-c_2}$, where the randomness is taken over the random partition of $S_k \cup S_{k+1} \cup \cdots \cup S_r$.*

Proof. For convenience, let $T = S_k \cup \cdots \cup S_r$ and $d' = \text{co-dim}(H_T)$. Suppose the hypercube H_T is parameterized by $S_T \subseteq T$ and $y_T \in \{0,1\}^T$, i.e., $H_T = \{x \in \{0,1\}^T \mid x_T = y_T\}$. Because $\text{co-dim}(H_T) = |S_T| \le |T|$, we have $d' \le m(r-k+1)$. For each $\ell \in T$, let Z_ℓ denote the indicator whether the variable ℓ is in S_i. Because $|T| = m(r-k+1)$ and $S_k, S_{k+1}, \cdots, S_r$ is a uniform partition of T, for any $\ell \in T$, the probability that $Z_\ell = 1$ is $\frac{1}{r-k+1}$. In addition, variables in $\{Z_\ell\}_{\ell \in T}$ follow a permutation distribution and are thus negatively associated.

Let $Z = \sum_{\ell \in S_T} Z_\ell$ denote the number of variables in $S_i \cap S_T$. It is clear that $\mathbb{E}[Z] = \frac{d'}{r-k+1}$. According to the definition of codimension, we have $Z = \text{co-dim}(H_{S_i})$. Because of the Chernoff bound and $d' \le m(r-k+1)$, for any $c_2 > 0$,

$$\mathbb{P}\left[\left|Z - \frac{d'}{r-k+1}\right| > \sqrt{3c_2 m \log n}\right] \le 2\exp\left(-\frac{3c_2 m \log n}{3d'/(r-k+1)}\right) \le 2\exp(-c_2 \log n) = 2n^{-c_2}.$$

Therefore, the probability of $\text{co-dim}(H_{S_i})$ being $d'/(r-k+1) \pm \sqrt{3c_2 m \log n}$ is at least $1 - 2n^{-c_2}$. □

Putting the previous lemmas together, we obtain the following key lemma for the hardness of parallel search via counting. If we only reveal the information about the first blocks $k-1$ (i.e., the partition $S_1, \cdots, S_{k-1}$ and the parameters to define the true strings $B_1, v_1, \cdots, B_{k-1}, v_{k-1}$), the return value of any query is determined solely by the information about the first k blocks with high

probability. This lemma implies that without any information about the k-th block and its subsequent blocks, any algorithm that uses one round of queries can only learn about the information in the k-th block (with high probability).

Lemma 36. *Fix any $k \in [r-1]$, any hypercube H and any realization of $S_1, B_1, v_1, \cdots, S_{k-1}, B_{k-1}, v_{k-1}$. With probability at least $1 - 3n^{-(c+5/3)}$,*

$$\mu(H) = \left(\prod_{i\in[k]} \mu_i(H_{S_i})\right) \cdot 2^{-\text{co-dim}(H)+\text{co-dim}(H_{S_1\cup S_2\cup\cdots\cup S_k})+\sum_{i>k} a_i}. \tag{6}$$

where the probability is taken over the random partition of $S_k \cup S_{k+1} \cup \cdots \cup S_r$ and the randomness of $B_k, v_k, B_{k+1}, v_{k+1}, \cdots, B_r, v_r$.

Proof. According to Lemma 32, it suffices to show that with probability $1 - n^{-(c+5/3)}$, we have

$$\prod_{i=k}^{r} \mu_i(H_{S_i}) = \mu_k(H_{S_k}) \cdot 2^{-\text{co-dim}(H)+\text{co-dim}(H_{S_1\cup S_2\cup\cdots\cup S_k})+\sum_{i>k} a_i}. \tag{7}$$

Let $d' = \text{co-dim}(H_{S_k\cup S_{k+1}\cup\cdots\cup S_r})$. Next, we prove the lemma by discussing two cases: $d'/(r-k+1) \geq \frac{a_{k-1}+a_k}{2}$ and $d'/(r-k+1) < \frac{a_{k-1}+a_k}{2}$. Before the discussion, recall that we define $m = 4n^{2/3}((c+2)\log n)^{1/3}$ and $a_i = i \cdot 12n^{1/3}((c+2)\log n)^{2/3}$ for each $i \in [r]$.

Case #1: when $d'/(r-k+1) \geq \frac{a_k+a_{k+1}}{2}$. According to the definition of a_k and a_{k+1}, $d'/(r-k+1) \geq a_k + 6n^{1/3}((c+2)\log n)^{2/3}$. Because of Lemma 35, with probability at least $1 - 2n^{-(c+2)}$,

$$\begin{aligned}\text{co-dim}(H_{S_k}) \geq \frac{d'}{r-k+1} - \sqrt{3(c+2)m\log n} \geq \\ a_k + 6n^{1/3}((c+2)\log n)^{2/3} - 2\sqrt{3}n^{1/3}((c+2)\log n)^{2/3} > \\ a_k + n^{1/3}((c+2)\log n)^{2/3} > a_k + (c+2)\log n.\end{aligned}$$

Because of Lemma 33, supposing $\text{co-dim}(H_{S_k}) > a_k + (c+2)\log n$, we have $\mu_k(H_{S_k}) = 0$ with probability at least $1 - n^{-(c+2)}$. Therefore, with probability at least $1 - 3n^{-(c+2)}$, both the LHS and RHS of Eq. (7) equal 0.

Case #2: when $d'/(r-k+1) < \frac{a_k+a_{k+1}}{2}$. According to the definition of a_k and a_{k+1}, $d'/(r-k+1) < a_{k+1} - 6n^{1/3}((c+2)\log n)^{2/3}$. Because of Lemma 35 and the union bound, with probability at least $1 - 2n^{-(c+5/3)}$, for all $j > k$,

$$\begin{aligned}\text{co-dim}(H_{S_j}) \leq \frac{d'}{r-k+1} + \sqrt{3(c+2)m\log n} \leq \\ a_k - 6n^{1/3}((c+2)\log n)^{2/3} + 2\sqrt{3}n^{1/3}((c+2)\log n)^{2/3} < \\ a_j - n^{1/3}((c+2)\log n)^{2/3} < a_j - (c+2)\log n.\end{aligned}$$

Because of Lemma 33, supposing $\text{co-dim}(H_{S_j}) < a_j - (c+2)\log n$ for any $j > k$, we have $\mu_j(H_{S_j}) = 2^{a_j - \text{co-dim}(H_{S_j})}$ for any $j > k$ with probability at least $1 - n^{-(c+2)}$. Therefore, with probability at least

$1 - 3n^{-(c+5/3)}$, we have

$$\prod_{i=k}^{r} \mu_i(H_{S_i}) \quad = \quad \mu_k(H_{S_k}) \cdot \prod_{j=k+1}^{r} 2^{a_j - \text{co-dim}(H_{S_j})} \quad = \quad \mu_k(H_{S_k}) \cdot 2^{\sum_{j>k} a_j - \sum_{j>k} \text{co-dim}(H_{S_j})}.$$

Since $S_1, S_2, \cdots, S_r$ is a partition, we have $\sum_{j>k} \text{co-dim}(H_{S_j}) = \text{co-dim}(H_{S_{k+1} \cup \cdots \cup S_r}) = \text{co-dim}(H) - \text{co-dim}(H_{S_1 \cup \cdots \cup S_k})$. Hence, we obtain Eq. (7) for this case. □

Finally, we can establish the main theorem of this section.

Proof of Theorem 31. We show the following statement by induction: for any $i \in [r]$, given the sets $S_1, S_2, \cdots, S_{i-1}$ and information $B_1, v_1, \cdots, B_{i-1}, v_{i-1}$, any deterministic algorithm can only find a solution x with probability at most $3(r - i + 1)n^{-5/3}$ after $r - i$ rounds. Note that given sets $S_1, S_2, \cdots, S_{i-1}$ and $B_1, v_1, \cdots, B_{i-1}, v_{i-1}$, the remaining sets $S_i, \cdots, S_r$ form a uniform random partition of $[n] \setminus (S_1 \cup \cdots \cup S_{i-1})$ and the remaining randomness $B_i, v_i, \cdots, B_r, v_r$ are uniformly at random. When $i = 1$, the statement is equivalent to any deterministic algorithm cannot find a solution with probability $n^{-4/3}$ after r rounds of queries. According to Yao's minimax principle, this implies that, for any randomized algorithm, there exists an instance such that the algorithm can only find a solution with probability at most $n^{-4/3}$ after $r-1$ rounds of queries. Next, we prove these statements to finish our proof.

The base case is $i = r$. With no queries, any deterministic algorithm returns a fixed x^*. Since $v_r \in \mathbb{F}_2^{m-a_r}$ is uniformly random, $m = 4n^{2/3}((c+2)\log n)^{1/3}$ and $a_r = 3n^{2/3}((c+2)\log n)^{1/3}$, the probability that x^* is a solution is $\mathbb{P}[B_r x^* = v_r] < 2^{a_i - m} < n^{-5/3}$.

Suppose we have shown for $i = k+1$ (where $1 \le k \le r-1$). Consider any deterministic algorithm ALG. Given $S_1, S_2, \cdots, S_{k-1}$ and $B_1, v_1, \cdots, B_{k-1}, v_{k-1}$, when ALG finds a solution x^* in $r - k$ rounds, at least one of the following events occurs.

- There exists an ALG's first-round query such that its return value does not follow Eq. (6).
- Given $S_1, S_2, \cdots, S_k$ and $B_1, v_1, \cdots, B_k, v_k$, simulating the round 1 queries of ALG by Eq. (6) and running ALG from round 2, we can find a solution x^* in $r - k - 1$ rounds.

In the first round of the algorithm, according to Lemma 36, with probability at least $1 - 3n^{-5/3}$, the return values of all the n^c queries follow Eq. (6). Therefore, the probability that the first event occurs is at most $3n^{-5/3}$. Also, note that Eq. (6) is fully determined by the the information in the first k blocks – $S_1, S_2, \cdots, S_k$ in the partition and the bits $B_1, v_1, \cdots, B_k, v_k$ used in the construction of the first k random linear codes. We can correctly simulate the first-round queries in the second event. Due to the induction hypothesis, the second event occurs with probability at most $3(r-k)n^{-5/3}$. Because of the union bound, ALG can find a solution with probability at most $3(r-k+1)n^{5/3}$ using $r - k$ rounds of queries. □

6 Lower bound of Algorithm 2

In this section, we show that our analysis of Algorithm 2 is tight up to polylogarithmic factors.

Theorem 37. *There exists an instance such that Algorithm 2 terminates in $\Omega(\frac{n^{2/3}}{\log n})$ rounds with probability at least 0.99.*

The proof of this lower bound uses the explicit construction of the universal coupler used in the algorithm (i.e. MinCoupler). We are unaware whether this lower bound holds if we use any universal coupler in the algorithm.

Errors of the universal coupler. Given a distribution μ, we identify a set of randomness r where MinCoupler produces different samples with constant probability for μ and υ when the second distribution υ is very different from μ. Here, we say υ is very different from μ, if we sample x from μ, there is a constant probability of having $\upsilon(x) < (1-\delta)\mu(x)$ for some reasonably large δ. Recall the construction of MinCoupler – we encode r as pairs $(x_1, p_1), (x_2, p_2), \cdots \in [q] \times [0,1]$, choose the minimum index i^* such that $p_{i^*} \le \mu(x_{i^*})$ and let x_{i^*} be the output of the universal coupler. The output x_{i^*} follows the distribution μ. If we consider the restricted set of r such that $p_{i^*} \ge (1-\delta)\mu(x_{i^*})$, for any distribution υ that is very different from μ, MinCoupler will produce different samples for μ and υ with constant probability. We formalize the above argument as the following lemma, whose proof is deferred to Appendix A.3.

Lemma 38 (Sure mistakes made by MinCoupler)**.** *Consider any distribution $\mu \in \Delta_q$ and any $\delta > 0$. Suppose $\mathcal{R}(\mu, \delta)$ is the set of randomness $r = ((x_1, p_1), (x_2, p_2), \cdots)$ used by MinCoupler such that $p_{i^*} \ge (1-\delta)\mu(x_{i^*})$, where $i^* = \min\{i : p_i \le \mu(x_i)\}$ is the index chosen by MinCoupler.*

Then, for any distribution $\upsilon \in \Delta_q$, we have

$$\mathbb{P}_{r\sim\mathcal{R}(\mu,\delta)}\left[\textsc{MinCoupler}(\mu, r) \neq \textsc{MinCoupler}(\upsilon, r)\right] \ge \frac{1-\upsilon_{\max}}{2} \cdot \mathbb{P}_{x\sim\mu}[\upsilon(x) < (1-\delta)\mu(x)],$$

where $\upsilon_{\max}$ is the maximum mass $\max_{x\in[q]} \upsilon(x)$ of the distribution υ.

The (random) hard instances. Suppose $m = 20n$. Consider parameters $y_1, y_2, \cdots, y_n \in \mathbb{R}^m$, which are randomly constructed in symmetry and will be stated later. Let $z \sim N(0, I_m)$ be a random vector. For each $i \in [n]$, let the variable $X_i = \mathsf{round}(\langle y_i, z\rangle)$, where the rounding function $\mathsf{round}(x)$ is constructed as follows.

$$\mathsf{round}(x) = \min\left\{n^4 \log n, \max\left\{-n^4 \log n, \left\lfloor n^4 x \right\rfloor\right\}\right\}.$$

To prove the lower bound, we consider the parameters $y_1, y_2, \cdots, y_n$ as i.i.d.s following the distribution $N(0, \frac{1}{m} I_m)$ and use $V_i = \langle y_i, z\rangle$ to denote the variables before rounding. It can be shown that, with high probability, X_i equals the floor of $n^4 V_i$ for any $i \in [n]$.

Lemma 39. *For any $y_1, y_2, \cdots, y_n$, with probability at least $1 - n^{-\Omega(\log n)}$, $\forall i \in [n]$, $X_i = \left\lfloor n^4 V_i \right\rfloor$.*

In addition, $\|y_i\|^2$ follows the χ^2_m distribution (with a $\frac{1}{m}$ factor). We can use the following to show that $\|y_i\|^2$ is concentrated within $1 \pm \epsilon$ with probability at least $1 - 2^{-n^{\Omega(1)}}$ for any constant $\epsilon > 0$.

Lemma 40 (Laurent-Massart bound [Lemma 1, LM00])**.** *Let $y \sim N(0, I_m)$ and $a \in \mathbb{R}^m_{\ge 0}$. Let $Z = \sum_{i\in[m]} a_i(y_i^2 - 1)$. Then, for any $x \ge 0$,*

$$\mathbb{P}\left[|Z| > 2\|a\|_2\sqrt{x} + 2\|a\|_\infty x\right] \le 2\exp(-x).$$

Suppose $y'_1, \cdots, y'_n$ are the vectors generated by the Gram-Schmidt orthogonalization procedure on $y_1, \cdots, y_n$. With this lemma and the fact that $y_1, \cdots, y_n$ are i.i.d. Gaussians, if we write each y'_i as a linear expression of $y_1, \cdots, y_n$, the coefficients can be bounded polynomially in n. The proof is deferred to Appendix A.4.

Lemma 41. *Suppose $n \le m/20$ and $y_1, y_2, \cdots, y_n$ are i.i.d. vectors following the distribution $N(0, \frac{1}{m} I_m)$. Consider $y'_1, y'_2, \cdots, y'_n$ as the vectors generated by the Gram-Schmidt orthogonalization procedure on the vectors $y_1, y_2, \cdots, y_n$. Suppose $\forall j \in [n], y'_j = \sum_{k=1}^{j} c_{jk} y_k$. With probability at least $1 - O(n^{-3})$, for any $\ell \in [n]$, $\sum_{j=1}^{n} c_{j\ell}^2 \le 2n$.*

Next, we show that if $i - a = \Omega(n^{1/3})$, the conditional distribution of $X_i | \{X_j\}_{j\in[a]}$ and $X_i | \{X_j\}_{j\in[i-1]}$ can be very different under random conditioning of $X_1, \cdots, X_{i-1}$ for $\delta = \Omega(n^{-1/3})$. We formalize it in Lemma 42. Its proof can be summarized by the following 3 key ingredients:

1. $V_i | \{V_j\}_{j\in[a]}$ and $V_i | \{V_j\}_{j\in[i-1]}$ can be neatly expressed as (randomly constructed) normal distributions whose means have a difference $\Omega(n^{-1/3})$ with a constant probability and whose variances are both $\Omega(1)$ with a high probability.
2. Rounding two normal distributions that satisfy the above two properties gives two very different discrete distributions for $\delta = \Omega(n^{-1/3})$, implying that $X_i | \{V_j\}_{j\in[a]}$ and $X_i | \{V_j\}_{j\in[i-1]}$ are very different.
3. Using Lemma 41 to handle random noise produced by conditioning on $\{X_j\}$ instead of $\{V_j\}$.

The full proof is deferred to Appendix A.5.

Lemma 42. *Consider any $a, i \in [n]$ such that $i - a > 40n^{1/3}$. Suppose μ, ν are the (randomly constructed) distributions of $X_i | \{X_j\}_{j\in[a]}$ and $X_i | \{X_j\}_{j\in[i-1]}$. There exist constants $c_1 > 0, c_2 \in (0,1)$ such that*

$$\mathbb{E}_{y_1,\cdots,y_n} \mathbb{E}_{X_1,X_2,\cdots,X_{i-1}} \left[(1 - \nu_{\max}) \cdot \mathbb{P}_{k\sim\mu} \left[\frac{\nu(k)}{\mu(k)} \le 1 - c_1 \cdot n^{-1/3} \right] \right] \ge c_2.$$

$\widetilde{\Omega}(n^{2/3})$ lower bound of Algorithm 2. In the rest of this section, we suppose c_1, c_2 are the constants stated in Lemma 42. We can show in Lemma 43, the algorithm only makes small progress (i.e., a increases by $O(n^{1/3})$) with probability at least $\Omega(1)$ in each round before termination.

Lemma 43. *Suppose $\forall i \in [n], \sigma(i) = i$. For any $a_0 \in [n-1] \cup \{0\}$, if we initiate Algorithm 2 with $a = a_0$ and $\forall i \in [a_0], x_i^0 = \tilde{x}_i(\sigma, u)$, the probability that the algorithm will have $a \le a_0 + (40 + c_1^{-1} \cdot \lceil \log(4c_2^{-1}) \rceil) n^{1/3}$ after one round is at least $\frac{c_2}{4}$, where the probability is taken over the randomness $y_1, y_2, \cdots, y_n$ used in the constructions of the instances and the randomness $u_1, \cdots, u_n$ used by the algorithm.*

Proof. Let $obj = a_0 + (40 + c_1^{-1} \cdot \lceil \log(4c_2^{-1}) \rceil) n^{1/3}$ be the objective value of a after one round. Let $\mathcal{I}$ be the set of integers in $[a_0 + 40n^{1/3}, obj]$. For each $i > a$, let μ_i be the distribution of X_i conditioning on $\{X_j = \tilde{x}_j(\sigma, u)\}_{j\in[a]}$ and let ν_i be the distribution of X_i conditioning on $\{X_j = \tilde{x}_j(\sigma, u)\}_{j\in[i-1]}$. Let $\mathcal{A}_i$ be the event that $\forall i < j \in \mathcal{I}, u_j \notin \mathcal{R}(\mu_j, c_1 \cdot n^{-1/3})$ and $u_i \in \mathcal{R}(\mu_j, c_1 \cdot n^{-1/3})$. It is clear that $\mathcal{A}_i$ are disjoint events. Note that we have $a > obj$ only if for any $i \in \mathcal{I}$, $\textsc{MinCoupler}(\mu_i, u_i) =$

$\textsc{MinCoupler}(v_i, u_i)$. Therefore, we have

$$\begin{aligned}
\mathbb{P}[a > obj] &\le \mathbb{P}[\forall i \in \mathcal{I}, \textsc{MinCoupler}(\mu_i, u_i) = \textsc{MinCoupler}(v_i, u_i)] \\
&= 1 - \mathbb{P}[\exists i \in \mathcal{I}, \textsc{MinCoupler}(\mu_i, u_i) \neq \textsc{MinCoupler}(v_i, u_i)] \\
&\le 1 - \mathbb{P}[\exists i \in \mathcal{I}, u_i \in \mathcal{R}(\mu_i, c_1 \cdot n^{-1/3}) \text{ and } \textsc{MinCoupler}(\mu_i, u_i) \neq \textsc{MinCoupler}(v_i, u_i)] \\
&\le 1 - \sum_{i \in \mathcal{I}} \mathbb{P}[\mathcal{A}_i] \cdot \mathbb{P}\left[\textsc{MinCoupler}(\mu_i, u_i) \neq \textsc{MinCoupler}(v_i, u_i) \,\middle|\, \mathcal{A}_i\right] \\
&\le 1 - \left(\sum_{i \in \mathcal{I}} \mathbb{P}[\mathcal{A}_i]\right) \cdot \min_{i \in \mathcal{I}} \mathbb{P}\left[\textsc{MinCoupler}(\mu_i, u_i) \neq \textsc{MinCoupler}(v_i, u_i) \,\middle|\, \mathcal{A}_i\right].
\end{aligned}$$

Since $\mathcal{A}_i$ are disjoint events, $\sum_{i \in \mathcal{I}} \mathbb{P}[\mathcal{A}_i]$ equals the probability that there exists $i \in \mathcal{I}$ such that $u_i \in \mathcal{R}(\mu_i, c_1 \cdot n^{-1/3})$. Because $u_i, i \in \mathcal{I}$ are i.i.d.s in $[0,1]$ and $|\mathcal{I}| = c_1^{-1}\lceil \log(4c_2^{-1})\rceil n^{1/3}$, this sum of probabilities be lower bounded as follows:

$$\sum_{i \in \mathcal{I}} \mathbb{P}[\mathcal{A}_i] = \mathbb{P}\left[\exists i \in \mathcal{I}, u_i \in \mathcal{R}\left(\mu_i, c_1 \cdot n^{-1/3}\right)\right] \ge 1 - \left(1 - c_1 \cdot n^{-1/3}\right)^{c_1^{-1}\log(4c_2^{-1})n^{1/3}} \ge 1 - \frac{c_2}{4}.$$

On the other hand, for any $i \in \mathcal{I}$, conditioning on the event $\mathcal{A}_i$, we have $u_i \sim \mathcal{R}(\mu_i, c_1 \cdot n^{-1/3})$ and $u_1, \cdots, u_{i-1}$ are uniform i.i.d.s in $[0,1]$. Therefore, $\tilde{x}_1(\sigma, u), \cdots, \tilde{x}_{i-1}(\sigma, u)$ follows its original marginal distribution of $X_1, \cdots, X_{i-1}$ after conditioning on $\mathcal{A}_i$. For any choice of $i \in \mathcal{I}$, we have the following lower bound:

$$\begin{aligned}
&\mathbb{P}\left[\textsc{MinCoupler}(\mu_i, u_i) \neq \textsc{MinCoupler}(v_i, u_i) \,\middle|\, \mathcal{A}_i\right] \\
=&\mathbb{E}_{y_1, \cdots y_n} \mathbb{E}_{u_1, \cdots, u_{i-1}}\left[\mathbb{P}_{u_i \sim \mathcal{R}(\mu_i, c_1 \cdot n^{-1/3})}\left[\textsc{MinCoupler}(\mu_i, u_i) \neq \textsc{MinCoupler}(v_i, u_i)\right]\right] \\
\ge&\mathbb{E}_{y_1, \cdots y_n} \mathbb{E}_{u_1, \cdots, u_{i-1}}\left[\frac{1 - (v_i)_{\max}}{2} \cdot \mathbb{P}_{x \sim \mu_i}\left[v_i(x) \le (1 - c_1 \cdot n^{-1/3})\mu_i(x)\right]\right] \ge \frac{c_2}{2}.
\end{aligned}$$

Therefore, $\mathbb{P}[a > obj] \ge 1 - (1 - \frac{c_2}{4}) \cdot \frac{c_2}{2} \ge 1 - \frac{c_2}{4}$. □

However, this constant probability does not suffice to show the $\widetilde{\Omega}(n^{2/3})$ lower bound. This is because we have not eliminated the possibility that the algorithm can terminate with a constant probability in each round. Next, we boost this probability of small progress to $1 - n^{-\Omega(1)}$ by constructing a new instance with $\mathrm{poly}\log(n)$ i.i.d. such instances. More specifically, let $g = 20c_2^{-1}\log n$ be the number of groups. The parameters of a new instance are the vectors $y_1, y_2, \cdots, y_n \in \mathbb{R}^m$ and the group numbers $h(1), h(2), \cdots, h(n) \in [g]$. Let $z_1, z_2, \cdots, z_g \sim N(0, I_m)$ be i.i.d. random Gaussian vectors. Then, the variables $X_1, \cdots, X_n$ in the new instance are defined as follows:

$$X_i = \mathsf{round}\left(\langle y_i, z_{h(i)}\rangle\right).$$

With this new construction, we can show that the algorithm will have small progress in each round with high probability.

Lemma 44. *Suppose $\forall i \in [n], \sigma(i) = i$. For any $a_0 \in [n-1] \cup \{0\}$, if we initiate Algorithm 2 with $a = a_0$ and $\forall i \in [a_0], x_i^0 = \tilde{x}_i(\sigma, u)$, the probability that the algorithm will have $a \ge a_0 + 2(40 + c_1^{-1} \cdot \lceil \log(4c_2^{-1})\rceil) n^{1/3} g$ after one round is at most $(1 + o(1))n^{-5}$, where the probability is taken over the randomness $y_1, y_2, \cdots, y_n, h(1), \cdots, h(n)$ used in constructions of the instances and the randomness $u_1, \cdots, u_n$ used by the algorithm.*

Proof. For convenience, we use $obj = a_0 + 2(40 + c_1^{-1} \cdot \lceil \log(4c_2^{-1}) \rceil) n^{1/3} g$ to denote the objective position for the algorithm to reach after one round. Let $\mathcal{I}$ be the set of integers in $[a_0 + 1, obj]$. For any $h \in [g]$, let $\mathcal{I}'_h$ be the set of $i \in [obj]$ such that $h(i) = h$. With probability $1 - 2^{-n^{\Omega(1)}}$ over the choices of $\{h(i) \mid i \in \mathcal{I}\}$, for any $h \in [g]$, the size of $\{i \in \mathcal{I} \mid h(i) = h\}$ (i.e., $\mathcal{I} \cap \mathcal{I}'_h$) is at least $(40 + c_1^{-1} \cdot \lceil \log(4c_2^{-1}) \rceil) n^{1/3}$. If $a \geq obj$ after one round of Algorithm 2, there should be

$$\forall i \in \mathcal{I}, \quad \textsc{MinCoupler}\left(\left(X_i \,\middle|\, \{X_j\}_{j \in [a_0]}\right), u_i\right) = \textsc{MinCoupler}\left(\left(X_i \,\middle|\, \{X_j\}_{j \in [i-1]}\right), u_i\right) \tag{8}$$

Note that for any $i, j \in [obj]$ such that $h(i) \neq h(j)$, X_i, X_j are independent. Eq. (8) is equivalent to $\forall h \in [g]$,

$$\forall i \in \mathcal{I} \cap \mathcal{I}'_h, \ \textsc{MinCoupler}\left(\left(X_i \,\middle|\, \{X_j\}_{j \in [a_0] \cap \mathcal{I}'_h}\right), u_i\right) = \textsc{MinCoupler}\left(\left(X_i \,\middle|\, \{X_j\}_{j \in [i-1] \cap \mathcal{I}'_h}\right), u_i\right)$$

Because of Lemma 43, for each $h \in [g]$, it happens with probability at most $1 - \frac{c_2}{4}$. Since $u_1, \cdots, u_n$ are i.i.d.s, under these choices of $\{h(i) \mid i \in \mathcal{I}\}$, the probability that $a \geq obj$ after one round is at most

$$\left(1 - \frac{c_2}{4}\right)^g = n^{-5}.$$

According to the union bound, we complete the proof. □

Finally, we establish the main theorem of this section.

proof of Theorem 37. Let $R(X, \sigma, u)$ be the number of rounds of Algorithm 2 on variables X, using randomness σ, u. It suffices to show that

$$\mathbb{P}_{\sigma,u,y,h}\left[R(X, \sigma, u) \geq \frac{n^{2/3}}{40c_2^{-1}(40 + c_1^{-1} \cdot \lceil \log(4c_2^{-1}) \rceil) \log n}\right] \geq 0.99. \tag{9}$$

Since $y_1, h(1), \cdots, y_n, h(n)$ are constructed in symmetry, for any permutation $\sigma \in \mathcal{S}_n$, $X_{\sigma(1)}, \cdots, X_{\sigma(n)}$ are identically distributed as $X_1, \cdots, X_n$. Therefore, it suffices to show Eq. (9) assuming $\sigma(i) = i$ for any $i \in [n]$. According to Lemma 44 and the union bound, with probability at least $1 - n^{-4}$ over the choice of y, h, u, for any initialization of a, we can increase a by at most $40c_2^{-1}(40 + c_1^{-1} \cdot \lceil \log(4c_2^{-1}) \rceil) n^{1/3} \log n$. In this case, the round complexity of Algorithm 2 is at least

$$\frac{n^{2/3}}{40c_2^{-1}(40 + c_1^{-1} \cdot \lceil \log(4c_2^{-1}) \rceil) \log n},$$

and thus we obtain Eq. (9). □

References

[Ana+20] Nima Anari, Nathan Hu, Amin Saberi, and Aaron Schild. "Sampling arborescences in parallel". In: *arXiv preprint arXiv:2012.09502* (2020).

[Ana+23a] Nima Anari, Callum Burgess, Kevin Tian, and Thuy-Duong Vuong. "Quadratic Speedups in Parallel Sampling from Determinantal Distributions". In: *Proceedings of the 35th ACM Symposium on Parallelism in Algorithms and Architectures*. 2023, pp. 367–377.

[Ana+23b] Nima Anari, Yizhi Huang, Tianyu Liu, Thuy-Duong Vuong, Brian Xu, and Katherine Yu. "Parallel Discrete Sampling via Continuous Walks". In: *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*. 2023, pp. 103–116.

[Bar16] Alexander Barvinok. *Combinatorics and complexity of partition functions*. Vol. 30. Springer, 2016.

[Bro+20] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. "Language models are few-shot learners". In: *Advances in neural information processing systems* 33 (2020), pp. 1877–1901.

[BRS19] Eric Balkanski, Aviad Rubinstein, and Yaron Singer. "An exponential speedup in parallel running time for submodular maximization without loss in approximation". In: *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM. 2019, pp. 283–302.

[BS20] Eric Balkanski and Yaron Singer. "A lower bound for parallel submodular minimization". In: *Proccedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing, STOC 2020, Chicago, IL, USA, June 22-26, 2020*. ACM, 2020, pp. 130–139.

[CCK21] Deeparnab Chakrabarty, Yu Chen, and Sanjeev Khanna. "A Polynomial Lower Bound on the Number of Rounds for Parallel Submodular Function Minimization". In: *62nd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2021, Denver, CO, USA, February 7-10, 2022*. IEEE, 2021, pp. 37–48.

[Cha+22] Deeparnab Chakrabarty, Andrei Graur, Haotian Jiang, and Aaron Sidford. "Improved lower bounds for submodular function minimization". In: *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE. 2022, pp. 245–254.

[Che+23] Charlie Chen, Sebastian Borgeaud, Geoffrey Irving, Jean-Baptiste Lespiau, Laurent Sifre, and John Jumper. "Accelerating large language model decoding with speculative sampling". In: *arXiv preprint arXiv:2302.01318* (2023).

[Dev+18] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. "Bert: Pre-training of deep bidirectional transformers for language understanding". In: *arXiv preprint arXiv:1810.04805* (2018).

[DFK91] Martin Dyer, Alan Frieze, and Ravi Kannan. "A random polynomial-time algorithm for approximating the volume of convex bodies". In: *Journal of the ACM (JACM)* 38.1 (1991), pp. 1–17.

[FHY21] Weiming Feng, Thomas P Hayes, and Yitong Yin. "Distributed metropolis sampler with optimal parallelism". In: *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*. SIAM. 2021, pp. 2121–2140.

[GM87] Hillel Gazit and Gary L Miller. "A parallel algorithm for finding a separator in planar graphs". In: *28th Annual Symposium on Foundations of Computer Science (sfcs 1987)*. IEEE. 1987, pp. 238–248.

[JSV04] Mark Jerrum, Alistair Sinclair, and Eric Vigoda. "A polynomial-time approximation algorithm for the permanent of a matrix with nonnegative entries". In: *Journal of the ACM (JACM)* 51.4 (2004), pp. 671–697.

[JVV86] Mark R Jerrum, Leslie G Valiant, and Vijay V Vazirani. "Random generation of combinatorial structures from a uniform distribution". In: *Theoretical computer science* 43 (1986), pp. 169–188.

[KUW88] Richard M. Karp, Eli Upfal, and Avi Wigderson. "The Complexity of Parallel Search". In: *J. Comput. Syst. Sci.* 36.2 (1988), pp. 225–253.

[Lee23] Holden Lee. "Parallelising Glauber dynamics". In: *arXiv preprint arXiv:2307.07131* (2023).

[LKM23] Yaniv Leviathan, Matan Kalman, and Yossi Matias. "Fast inference from transformers via speculative decoding". In: *International Conference on Machine Learning*. PMLR. 2023, pp. 19274–19286.

[LLV20] Wenzheng Li, Paul Liu, and Jan Vondrák. "A polynomial lower bound on adaptive complexity of submodular maximization". In: *Proccedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing, STOC 2020, Chicago, IL, USA, June 22-26, 2020*. ACM, 2020, pp. 140–152.

[LM00] Beatrice Laurent and Pascal Massart. "Adaptive estimation of a quadratic functional by model selection". In: *Annals of statistics* (2000), pp. 1302–1338.

[LM11] Hugo Larochelle and Iain Murray. "The neural autoregressive distribution estimator". In: *Proceedings of the fourteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings. 2011, pp. 29–37.

[LY22] Hongyang Liu and Yitong Yin. "Simple parallel algorithms for single-site dynamics". In: *STOC '22: 54th Annual ACM SIGACT Symposium on Theory of Computing, Rome, Italy, June 20 - 24, 2022*. ACM, 2022, pp. 1431–1444.

[Mon08] Andrea Montanari. "Estimating random variables from random sparse observations". In: *European Transactions on Telecommunications* 19.4 (2008), pp. 385–403.

[MVV87] Ketan Mulmuley, Umesh V Vazirani, and Vijay V Vazirani. "Matching is as easy as matrix inversion". In: *Proceedings of the nineteenth annual ACM symposium on Theory of computing*. 1987, pp. 345–354.

[RT12] Prasad Raghavendra and Ning Tan. "Approximating CSPs with global cardinality constraints using SDP hierarchies". In: *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2012, Kyoto, Japan, January 17-19, 2012*. SIAM, 2012, pp. 373–387.

[RY13] Daniel Revuz and Marc Yor. *Continuous martingales and Brownian motion*. Vol. 293. Springer Science & Business Media, 2013.

[Shi+23] Andy Shih, Suneel Belkhale, Stefano Ermon, Dorsa Sadigh, and Nima Anari. "Parallel Sampling of Diffusion Models". In: *arXiv preprint arXiv:2305.16317* (2023).

[Son+21] Yang Song, Chenlin Meng, Renjie Liao, and Stefano Ermon. "Accelerating feedforward computation via parallel nonlinear equation solving". In: *International Conference on Machine Learning*. PMLR. 2021, pp. 9791–9800.

[SSE22] Andy Shih, Dorsa Sadigh, and Stefano Ermon. "Training and Inference on Any-Order Autoregressive Models the Right Way". In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 2762–2775.

[ŠVV09] Daniel Štefankovič, Santosh Vempala, and Eric Vigoda. "Adaptive simulated annealing: A near-optimal connection between sampling and counting". In: *Journal of the ACM (JACM)* 56.3 (2009), pp. 1–36.

[Ten95] Shang-Hua Teng. "Independent sets versus perfect matchings". In: *Theoretical Computer Science* 145.1-2 (1995), pp. 381–390.

[Vas+17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. "Attention is all you need". In: *Advances in neural information processing systems* 30 (2017).

[VKK16] Aäron Van Den Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. "Pixel recurrent neural networks". In: *International conference on machine learning*. PMLR. 2016, pp. 1747–1756.

[Wei06] Dror Weitz. "Counting independent sets up to the tree threshold". In: *Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*. 2006, pp. 140–149.

[WH20] Auke Wiggers and Emiel Hoogeboom. "Predictive sampling with forecasting autoregressive models". In: *International Conference on Machine Learning*. PMLR. 2020, pp. 10260–10269.

[Yan+19] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. "Xlnet: Generalized autoregressive pretraining for language understanding". In: *Advances in neural information processing systems* 32 (2019).

A Deferred Proofs

A.1 Proof of Lemma 27

We use x^t, y^t, a^t to denote the intermediate variables used in the algorithm with the exact counting oracle, and use $\hat{x}^t, \hat{y}^t, \hat{a}^t$ to denote the intermediate variables used in the algorithm with the approximate counting oracle. We prove the claim that $a^t = \hat{a}^t$ and $x^t_{\sigma(j)} = \hat{x}^t_{\sigma(j)} = \tilde{x}_j(\sigma, u)$ for any $j \in [a^t]$ after each round t we compute a^t.

When $t = 0$, $a^t = \hat{a}^t = 0$ and this claim is clearly true. Consider $t \geq 1$. Suppose we have proved the claim for $t-1$. Because of the induction hypothesis and the fact that (σ, u) is good, $y^t = \hat{y}^t$. Because of the definition of a^t and Lemma 19, we have $y^t_{\sigma(j)} = x^t_{\sigma(j)} = \tilde{x}_j(\sigma, u)$ for any $j \in [a^t - 1]$ and we have $y^t_{\sigma(a^t)} \neq x^t_{\sigma(a^t)} = \tilde{x}_{a^t}(\sigma, u)$. Because of $\hat{y}^t = y^t$ and the induction hypothesis, we have $\hat{x}^t_{\sigma(j)} = x^t_{\sigma(j)} = \tilde{x}_j(\sigma, u)$ for any $j \in [a^t]$. Therefore, according to the definition of $\hat{a}^t$, $\hat{a}^t = a^t$.

Finally, we show how to obtain the lemma using this claim. If the algorithm with the exact counting oracle terminates with $a^t = n$ in some round, the claim directly implies that the algorithm with the approximate counting oracle terminates in the same round and outputs the vector. Otherwise, suppose t is the round in which the algorithm with the exact counting oracle terminates. There is $y^t = x^t$. As discussed above, the claim gives $\hat{y}^t = y^t$. Therefore, $\hat{y}^t_{\sigma(i)} = x^t_{\sigma(i)} = \tilde{x}_i(\sigma, u)$ for any $i \in [n]$. Since (σ, u) is good, the algorithm with the approximate counting oracle will generate $\hat{x}^t = \hat{y}^t$ and terminate in this round. The output $\hat{x}^t$ is thus the same as x^t.

A.2 Proof of Lemma 28

Note that we use q queries of $\hat{\mu}$ to compute each $v_{i|a}(\sigma, u)$. With probability at least $1 - n^2 q\delta$, all queries $\hat{\mu}$ we use while computing $v_{i|a}(\sigma, u)$s satisfy Eq. (5). Under these circumstances, for any $0 \leq a < i \leq n$ and any $x \in [q]$,

$$\left(v_{i|a}(\sigma, u)\right)(x) \geq \frac{1-\epsilon}{1+\epsilon} \cdot \mathbb{P}\left[X_{\sigma(i)} = x \;\middle|\; \{X_{\sigma(j)} = \tilde{x}_j(\sigma, u)\}_{j \in [a]}\right].$$

Therefore, $d_{\mathrm{TV}}\left(X_{\sigma(i)} \;\middle|\; \{X_{\sigma(j)} = \tilde{x}_j(\sigma, u)\}_{j \in [a]}, v_{i|a}(\sigma, u)\right) \leq 1 - \frac{1-\epsilon}{1+\epsilon} \leq 2\epsilon$. Recall that the universal coupler guarantees $\mathbb{P}[\textsc{MinCoupler}(\mu, u) \neq \textsc{MinCoupler}(v, u)] \leq 2\, d_{\mathrm{TV}}(\mu, v)$ for any two distributions μ, v. Under these circumstances, due to the union bound, (σ, u) is good with probability at least $1 - 4n^2\epsilon$. Putting things together, any (σ, u) is good with probability at least $(1 - n^2 q\delta) \cdot (1 - 4n^2\epsilon) \geq 1 - O(n^2\epsilon + n^2 q\delta)$.

A.3 Proof of Lemma 38

Let $i^*_\mu = \min\{i \mid p_i \le \mu(x_i)\}$ and $i^*_\nu = \min\{i \mid p_i \le \nu(x_i)\}$. According to the definition of MinCoupler, $\textsc{MinCoupler}(\mu, r) \neq \textsc{MinCoupler}(\nu, r)$ if and only if $x_{i^*_\mu} \neq x_{i^*_\nu}$. Observe that

$$\mathbb{P}_{r\sim\mathcal{R}(\mu,\delta)}\left[x_{i^*_\mu} \neq x_{i^*_\nu}\right] = \frac{\mathbb{P}_{r\sim[0,1]}\left[i^*_\mu < i^*_\nu,\ r \in \mathcal{R}(\mu,\delta)\right] \cdot \mathbb{P}\left[x_{i^*_\mu} \neq x_{i^*_\nu} \,\middle|\, i^*_\mu < i^*_\nu,\ r \in \mathcal{R}(\mu,\delta)\right]}{\mathbb{P}_{r\sim[0,1]}\left[r \in \mathcal{R}(\mu,\delta)\right]}.$$

On the denominator, we have,

$$\begin{aligned}\mathbb{P}_{r\sim[0,1]}\left[r \in \mathcal{R}(\mu,\delta)\right] &= \sum_{i\ge 1} \mathbb{P}_{r\sim[0,1]}\left[i^*_\mu = i\right] \cdot \mathbb{P}_{r\sim[0,1]}\left[p_i \ge (1-\delta)\mu(x_i) \,\middle|\, p_i \le \mu(x_i)\right] \\ &= \sum_{i\ge 1} \mathbb{P}_{r\sim[0,1]}\left[i^*_\mu = i\right] \cdot \delta = \delta. \qquad (p_i \sim [0,1])\end{aligned}$$

On the other hand, let $i^*_{\min} = \min\{i^*_\mu, i^*_\nu\}$. As discussed in the preliminary, we have $i^*_{\min} = \min\{i \mid p_i \le \max\{\mu(x_i), \nu(x_i)\}\}$. For any $i \ge 1$,

$$\begin{aligned}&\mathbb{P}_{r\sim[0,1]}\left[i^*_\mu < i^*_\nu,\ r \in \mathcal{R}(\mu,\delta) \,\middle|\, i^*_{\min} = i\right] \\ =&\,\mathbb{P}_{r\sim[0,1]}\left[p_i > \nu(x_i),\ p_i \in \left[(1-\delta)\mu(x_i),\ \mu(x_i)\right] \,\middle|\, i^*_{\min} = i\right] \\ \ge&\,\mathbb{P}_{r\sim[0,1]}\left[\nu(x_i) < (1-\delta)\mu(x_i),\ p_i \in \left[(1-\delta)\mu(x_i),\ \mu(x_i)\right] \,\middle|\, i^*_{\min} = i\right] \\ =&\,\mathbb{P}_{r\sim[0,1]}\left[\nu(x_i) < (1-\delta)\mu(x_i),\ p_i \in \left[(1-\delta)\mu(x_i),\ \mu(x_i)\right] \,\middle|\, p_i \le \max\{\mu(x_i), \nu(x_i)\}\right] \\ &\qquad\qquad (\text{(x_i, p_i)s are drawn independently}) \\ =&\,\frac{\mathbb{P}_{r\sim[0,1]}\left[\nu(x_i) < (1-\delta)\mu(x_i),\ p_i \in \left[(1-\delta)\mu(x_i),\ \mu(x_i)\right]\right]}{\mathbb{P}\left[p_i \le \max\{\mu(x_i), \nu(x_i)\}\right]} \\ =&\,\frac{q^{-1}\delta\sum_{x\in[q]}\mu(x)}{q^{-1}\sum_{x\in[q]}\max\{\mu(x),\nu(x)\}} \cdot \mathbb{P}_{r\sim[0,1]}\left[\nu(x_i) < (1-\delta)\mu(x_i) \,\middle|\, p_i \in \left[(1-\delta)\mu(x_i),\ \mu(x_i)\right]\right] \\ &\qquad\qquad (x_i \sim [q], p_i \sim [0,1]) \\ =&\,\frac{\delta}{1+d_{\mathrm{TV}}(\mu,\nu)} \cdot \mathbb{P}_{x\sim\mu}\left[\nu(x) < (1-\delta)\mu(x)\right] \ge \frac{\delta}{2} \cdot \mathbb{P}_{x\sim\mu}\left[\nu(x) < (1-\delta)\mu(x)\right].\end{aligned}$$

Therefore, we have

$$\mathbb{P}_{r\sim[0,1]}[i^*_\mu < i^*_\nu, r \in \mathcal{R}(\mu,\delta)] \ge \frac{\delta}{2} \cdot \mathbb{P}_{x\sim\mu}[\nu(x) < (1-\delta)\mu(x)].$$

In addition, for any $i \ge 1$, the event that $i = i^*_\mu < i^*_\nu$ and $r \in \mathcal{R}(\mu,\delta)$ is independent of the value of any x_j for $j > i$. Since for any $j \ge 1$ and any (possibly random) $x \in [q]$, $\mathbb{P}[x_j \neq x] \ge 1 - \nu_{\max}$, we have

$$\mathbb{P}[x_{i^*_\mu} \neq x_{i^*_\nu} \mid i^*_\mu < i^*_\nu, r \in \mathcal{R}(\mu,\delta)] \ge 1 - \nu_{\max}.$$

Therefore, on the numerator, we have,

$$\mathbb{P}_{r\sim\mathcal{R}(\mu,\delta)}\left[x_{i^*_\mu} \neq x_{i^*_\nu}\right] \ge \frac{\delta(1-\nu_{\max})}{2} \cdot \mathbb{P}_{x\sim\mu}\left[\nu(x) < (1-\delta)\mu(x)\right].$$

Putting things together, we obtain that $\mathbb{P}_{r\sim\mathcal{R}(\mu,\delta)}[x_{i^*_\mu} \neq x_{i^*_\nu}] \ge \frac{1-\nu_{\max}}{2} \cdot \mathbb{P}_{x\sim\mu}\left[\nu(x) < (1-\delta)\mu(x)\right]$ and thus the lemma.

A.4 Proof of Lemma 41

We shall prove this lemma by induction on i of the following statements: with probability at least $1 - i \cdot O(n^{-4})$, we have

$$\forall \ell \in [i], \quad \sum_{j=1}^{i} c_{j\ell}^2 \le 2 \cdot \left(1 + \frac{12 \log n}{m}\right)^i \tag{10}$$

When $i = 1$, Gram-Schmidt procedure gives $y_1' = y_1 / \|y_1\|_2$. Because of the Laurent-Massart bound, $\|y_1\|_2^2 \in 1 \pm 0.1$ with probability at least $1 - 2^{-n^{\Omega(1)}}$. Therefore, $c_{11}^2 \le 1.2$ with probability $1 - 2^{-n^{\Omega(1)}}$.

Suppose $k > 1$ and we have shown the statements for any $i < k$. Next, we show the statements for $i = k$. We only consider the randomness of y_k and consider the cases where Eq. (10) holds and $c_{jj}^2 \le 1.2$ holds for any $j \in [i]$. According to the Gram-Schmidt procedure, y_k' is obtained as follows: $y_k'' = y_k - \sum_{j \in [k-1]} \langle y_k, y_j' \rangle y_j'$ and $y_k' = y_k'' / \|y_k''\|_2$. Since $y_k \sim N(0, \frac{1}{m} I_m)$ and $\{y_j'\}_{j \in [k-1]}$ are orthonormal, $\langle y_k, y_j' \rangle$ are i.i.d.s following $N(0, \frac{1}{m})$. Suppose $y_k'' = y_k + \sum_{j \in [k-1]} c_{kj}' y_j$. Because we have

$$\sum_{j=1}^{k-1} \langle y_k, y_j' \rangle y_j' = \sum_{j=1}^{k-1} \langle y_k, y_j' \rangle \cdot \sum_{\ell=1}^{j} c_{j\ell} y_\ell = \sum_{\ell=1}^{k-1} \sum_{j=\ell}^{k-1} c_{j\ell} \langle y_k, y_j' \rangle y_\ell,$$

we have $c_{kk}' = 1$ and $c_{k\ell}' \sim N\left(0, \frac{1}{m} \sum_{j=\ell}^{k-1} c_{j\ell}^2\right)$ for any $\ell < k$. Therefore, for any $\ell \in [k-1]$, with probability at least $1 - O(n^{-5})$,

$$c_{k\ell}'^2 \le \frac{10 \log n}{m} \cdot \sum_{j=\ell}^{k-1} c_{j\ell}^2 \le \frac{10 \log n}{m} \cdot 2\left(1 + \frac{12 \log n}{m}\right)^{k-1}$$

Then, according to the second step of obtaining y_k', we have $c_{k\ell}^2 = c_{k\ell}'^2 / \|y_k''\|_2^2$ and $c_{kk}^2 = 1 / \|y_k''\|_2^2$. Note that $\|y_k''\|_2^2 = \|y_k\|_2^2 - \sum_{j \in [k-1]} (\langle y_k, y_j' \rangle)^2$. Since $\langle y_k, y_j' \rangle \sim N(0, \frac{1}{m})$ and $k < m/20$, we have $\|y_k''\|_2^2 \ge 0.9$ with probability at least $1 - 2^{-n^{\Omega(1)}}$. Therefore, with probability at least $1 - 2^{-n^{\Omega(1)}}$, $c_{kk}^2 \le 2$, and with probability at least $1 - O(n^{-4})$, for any $\ell \in [k-1]$, we have

$$c_{k\ell}^2 \le c_{k\ell}'^2 / 0.9 \le \frac{12 \log n}{m} \cdot 2\left(1 + \frac{12 \log n}{m}\right)^{k-1}.$$

Since Eq. (10) for $i = k-1$ holds with probability at least $1 - (k-1)O(n^{-4})$, we obtain the proof for $i = k$.

Finally, because $n \le m/20$, for any $\ell \in [n]$, we have

$$\sum_{j \in [n]} c_{j\ell}^2 \le 2 \cdot \left(1 + \frac{12 \log n}{m}\right)^n \le 2 \cdot \left(1 + \frac{12 \log n}{m}\right)^{m/20} \le 2n.$$

A.5 Proof of Lemma 42

Suppose $y_1', y_2', \cdots, y_n'$ are the vectors generated by the Gram-Schmidt orthogonalization procedure on $y_1, y_2, \cdots, y_n$. We assume that the conditions in Lemma 39 and Lemma 41 hold, which happens with probability $1 - O(n^{-3})$.

Consider we expand these vectors to an orthonormal basis $y'_1, y'_2, \cdots, y'_n, y'_{n+1}, \cdots, y'_m$ of $\mathbb{R}^m$. For any $i \in [n]$ and $0 \le j < i$, suppose $y_{i|j}$ is the projection of y_i on the linear span of $y_1, \cdots, y_j$ and let $y^{\perp}_{i|j} = y_i - y_{i|j}$. We have $y_{i|j} = \sum_{k=1}^{j} \langle y_i, y'_j \rangle y'_j$ and $y^{\perp}_{i|j} = \sum_{k=j+1}^{m} \langle y_i, y'_j \rangle y'_j$. We can rewrite $V_i = \langle y_{i|j}, z \rangle + \langle y^{\perp}_{i|j}, z \rangle$ for any $j < i$. Therefore, for any $a < i$, we can rewrite

$$\left(V_i \mid \{X_j\}_{j\in[i-1]}\right) = \left(\langle y_{i|a}, z\rangle \mid \{X_j\}_{j\in[a]}\right) + \left(\langle y_{i|i-1} - y_{i|a}, z\rangle \mid \{X_j\}_{j\in[i-1]}\right) + \langle y^{\perp}_{i|i-1}, z\rangle,$$
$$\left(V_i \mid \{X_j\}_{j\in[a]}\right) = \left(\langle y_{i|a}, z\rangle \mid \{X_j\}_{j\in[a]}\right) + \langle y^{\perp}_{i|a}, z\rangle.$$

Note that $\|y_{i|i-1} - y_{i|a}\|^2 = \sum_{k=a+1}^{i-1} (\langle y_i, y'_j \rangle)^2$. Because $y_i \sim N(0, \frac{1}{m} I_m)$, y'_j are orthonormal, and $y'_j (j < i)$s are independent with y_i, $\langle y_i, y'_j \rangle$ are i.i.d. variables following $N(0, \frac{1}{m})$. Therefore, according to the Laurent-Massart bound, $\|y_{i|i-1} - y_{i|a}\|^2 \in (1 \pm 0.01)^2 \cdot \frac{i-a-1}{m}$ with probability at least $1 - 2^{-n^{\Omega(1)}}$. This implies $\|y_{i|i-1} - y_{i|a}\| \ge 1.5 n^{-1/3}$ with probability at least $1 - 2^{-n^{\Omega(1)}}$. Since $n \le m/20$, we similarly have $\|y_{i|i-1}\|^2, \|y_{i|a}\|^2 \le 0.1$ and $\|y^{\perp}_{i|i-1}\|^2, \|y^{\perp}_{i|a}\|^2 \in [0.9, 1.1]$ with probability at least $1 - 2^{-n^{\Omega(1)}}$. We shall assume these conditions in the rest of the proof.

Because of our assumption (where the condition in Lemma 39 holds), $\forall i \in [n], V_i - X_i \in [0, n^{-4}]$. Observe that

$$\begin{aligned}
\langle y_{i|a}, z\rangle &= \sum_{j\in[a]} \langle y_i, y'_j\rangle \langle y'_j, z\rangle \\
&= \sum_{j\in[a]} \langle y_i, y'_j\rangle \sum_{k\in[j]} c_{jk} \langle y_k, z\rangle \\
&= \sum_{j\in[a]} \langle y_i, y'_j\rangle \sum_{k\in[j]} c_{jk} (X_k + (V_k - X_k)) && (V_k = \langle y_k, z\rangle) \\
&= \sum_{k\in[a]} \left(\sum_{j=k}^{a} c_{jk} \langle y_i, y'_j\rangle \right) \cdot (X_k + (V_k - X_k)) \\
&\stackrel{\text{def}}{=} \mathcal{E}_1 + \underbrace{\sum_{k\in[a]} \left(\sum_{j=k}^{a} c_{jk} \langle y_i, y'_j\rangle \right) \cdot X_k}_{B}
\end{aligned}$$

According to the Cauchy-Schwarz inequality and Lemma 41, all the coefficients $(\sum_{j=k}^{a} c_{jk} \langle y_i, y'_j \rangle)^2 \le (\sum_{j=1}^{n} c_{jk}^2)(\sum_{j=1}^{a} (\langle y_i, y'_j \rangle)^2) \le 2n^2 \|y_{i|a}\|^2$. According to our assumption of $\|y_{i|a}\|^2 \le 0.1$, $\mathcal{E}_1 \in \pm 2n^{-2}$.

On the other hand, we similarly have

$$\begin{aligned}
\langle y_{i|i-1} - y_{i|a}, z\rangle &= \sum_{k\in[i-1]} \left(\sum_{j=\max\{k,a+1\}}^{i-1} c_{jk} \langle y_i, y'_j\rangle \right) \cdot (X_k + (V_k - X_k)) \\
&\stackrel{\text{def}}{=} \mathcal{E}_2 + \underbrace{\sum_{k\in[i-1]} \left(\sum_{j=\max\{k,a+1\}}^{i-1} c_{jk} \langle y_i, y'_j\rangle \right) \cdot X_k}_{C},
\end{aligned}$$

and $\mathcal{E}_2 \in \pm 2n^{-2}$. Because $z \sim N(0, I_m)$, $\langle y_{i|i-1} - y_{i|a}, z\rangle \sim N(0, \|y_{i|i-1} - y_{i|a}\|^2)$, and we have $\mathbb{P}[\langle y_{i|i-1} - y_{i|a}, z\rangle \le -\|y_{i|i-1} - y_{i|a}\|] \ge 0.15$. Since $\mathcal{E}_2 \in \pm 2n^{-2}$ and $\|y_{i|i-1} - y_{i|a}\| \le -1.5n^{-1/3}$, we have $\mathbb{P}[C \le -n^{-1/3}] \ge 0.15$.

Note that conditioning on $\{X_j\}_{j\in[i-1]}$, B and C are fixed. Consider the cases where $C \le -n^{-1/3}$. Since $z \sim N(0, I_m)$, for any valid choice of $X_1, \cdots, X_{i-1}$, the distributions $\mu \sim (X_i \mid \{X_j\}_{j\in[a]})$ and $\nu \sim (X_i \mid \{X_j\}_{j\in[a]})$ can be concluded as follows:

1. μ is the value applying round on the sum of the fixed value B, random Gaussian $N(0, \|y_{i|a}^{\perp}\|^2)$ and random variables $(\mathcal{E}_1 \mid \{X_j\}_{j\in[a]}) \in \pm 2n^{-2}$.
2. ν is the value applying round on the sum of the fixed values B, C, random Gaussian $N(0, \|y_{i|i-1}^{\perp}\|^2)$ and random variables $(\mathcal{E}_1 \mid \{X_j\}_{j\in[a]}), (\mathcal{E}_2 \mid \{X_j\}_{j\in[i-1]}) \in \pm 2n^{-2}$.

Suppose f_1, f_2 are the density functions of $(\mathcal{E}_1 \mid \{X_j\}_{j\in[a]})$ and $(\mathcal{E}_2 \mid \{X_j\}_{j\in[i-1]})$. We have for any integer $k \in [-n^4 \log n, n^4 \log n]$,

$$\mu(k) = \int_{x=kn^{-4}}^{(k+1)n^{-4}} \int_{e_1} f_1(e_1) \cdot \frac{\exp(-(x - e_1 - B)^2/2\|y_{i|a}^{\perp}\|^2)}{\sqrt{2\pi\|y_{i|a}^{\perp}\|^2}} de_1 dx$$

$$\nu(k) = \int_{x=kn^{-4}}^{(k+1)n^{-4}} \int_{e_1,e_2} f_1(e_1)f_2(e_2) \cdot \frac{\exp(-(x - e_1 - e_2 - B - C)^2/2\|y_{i|i-1}^{\perp}\|^2)}{\sqrt{2\pi\|y_{i|i-1}^{\perp}\|^2}} de_1 de_2 dx$$

If $C < 0$, for any integer $k \in \{k \in \mathbb{Z} \mid kn^{-4} \in (4n^{-2} + B + 0.1, 4n^{-2} + B + 0.2)\} := \mathcal{K}$,

$$\mu(k) \ge \frac{\exp(-(kn^{-4} - B + 7n^{-2})^2/2\|y_{i|a}^{\perp}\|^2)}{\sqrt{2\pi}\|y_{i|a}^{\perp}\|} \cdot n^{-4}$$

$$\nu(k) \le \frac{\exp(-(kn^{-4} - B + |C|)^2/2\|y_{i|i-1}^{\perp}\|^2)}{\sqrt{2\pi}\|y_{i|i-1}^{\perp}\|} \cdot n^{-4}$$

In this case, $\nu(k)/\mu(k)$ is at most

$$\begin{aligned}
&\frac{\|y_{i|i-1}^{\perp}\|}{\|y_{i|a}^{\perp}\|} \cdot \exp\left(\frac{1}{2}\left(\frac{kn^{-4} - B + 7n^{-2}}{\|y_{i|a}^{\perp}\|} - \frac{kn^{-4} - B + |C|}{\|y_{i|i-1}^{\perp}\|}\right)\left(\frac{kn^{-4} - B + 7n^{-2}}{\|y_{i|a}^{\perp}\|} + \frac{kn^{-4} - B + |C|}{\|y_{i|i-1}^{\perp}\|}\right)\right) \\
=&\frac{\|y_{i|i-1}^{\perp}\|}{\|y_{i|a}^{\perp}\|} \cdot \exp\left(\Theta(1) \cdot \left(\frac{kn^{-4} - B + 7n^{-2}}{\|y_{i|a}^{\perp}\|} - \frac{kn^{-4} - B + 7n^{-2}}{\|y_{i|i-1}^{\perp}\|} - \frac{|C| - 7n^{-2}}{\|y_{i|i-1}^{\perp}\|}\right)\right) \\
\le& \exp\big(\Theta(-|C| + 7n^{-2})\big) = 1 - \Omega(n^{-1/3}) \qquad (\|y_{i|a}^{\perp}\| \ge \|y_{i|i-1}^{\perp}\|)
\end{aligned}$$

In addition, we have

$$\sum_{k\in\mathcal{K}} \mu(k) \ge \int_{0.1+7n^{-2}}^{0.2-7n^{-2}} \frac{\exp\left(-x^2/2\|y_{i|a}^{\perp}\|\right)}{\sqrt{2\pi}\|y_{i|a}^{\perp}\|} dx = \Omega(1).$$

Therefore, we conclude that $\mathbb{P}_{\nu\sim\mu}\left[\frac{\nu(k)}{\mu(k)} \le 1 - \Omega(n^{-1/3})\right] \ge \Omega(1)$ if $C \le -n^{-1/3}$. In addition, it is clear that $v_{\max} = O(n^{-4})$. Because $C \le -n^{-1/3}$ happens with a constant probability, we complete the proof.