Hidden Permutations to the Rescue: Multi-Pass Semi-Streaming Lower Bounds for Approximate Matchings

Sepehr Assadi* Janani Sundaresan†

October 12, 2023

Abstract

We prove that any semi-streaming algorithm for $(1-\varepsilon)$-approximation of maximum bipartite matching requires

$$\Omega\Big(\frac{\log(1/\varepsilon)}{\log(1/\beta)}\Big)$$

passes, where $\beta \in (0,1)$ is the largest parameter so that an n-vertex graph with n^β edge-disjoint induced matchings of size $\Theta(n)$ exist (such graphs are referred to as *Ruzsa-Szemerédi* graphs). Currently, it is known that

$$\Omega(\frac{1}{\log\log n}) \leqslant \beta \leqslant 1 - \Theta(\frac{\log^* n}{\log n})$$

and closing this huge gap between upper and lower bounds has remained a notoriously difficult problem in combinatorics.

Under the plausible hypothesis that $\beta = \Omega(1)$, our lower bound result provides the first **pass-approximation** lower bound for (small) **constant approximation** of matchings in the semi-streaming model, a longstanding open question in the graph streaming literature.

Our techniques are based on analyzing communication protocols for compressing (hidden) permutations. Prior work in this context relied on reducing such problems to Boolean domain and analyzing them via tools like XOR Lemmas and Fourier analysis on Boolean hypercube. In contrast, our main technical contribution is a hardness amplification result for permutations through concatenation in place of prior XOR Lemmas. This result is proven by analyzing permutations directly via simple tools from group representation theory combined with detailed information-theoretic arguments, and can be of independent interest.

*(sepehr@assadi.info) Cheriton School of Computer Science, University of Waterloo, and Department of Computer Science, Rutgers University. Supported in part by an Alfred P. Sloan Fellowship, a University of Waterloo startup grant, an NSF CAREER grant CCF-2047061, and a gift from Google Research.

†(jsundaresan@uwaterloo.ca) Cheriton School of Computer Science, University of Waterloo.

Contents

1 Introduction

In the semi-streaming model for graph computation, formalized by [FKM+05], the edges of an n-vertex graph $G = (V, E)$ are presented to the algorithm in some arbitrarily ordered stream. The algorithm can make one or few passes over this stream and uses $\widetilde{O}(n) := O(n \cdot \text{polylog}(n))$ memory to solve the given problem. The semi-streaming model has been at the forefront of research on processing massive graphs since its introduction almost two decades ago. In this work, we focus on the *maximum matching* problem in this model.

The maximum matching problem is arguably the most studied problem in the semi-streaming model and has been considered from numerous angles (this list is by no means a comprehensive summary of prior results):

- single-pass algorithms [FKM+05, GKK12, Kap13, Kap21, ABKL23],
- constant-pass algorithms [KMM12, EHM16, KT17, Kon18, KN21, FS22, KNS23, A22],
- $(1-\varepsilon)$-approximation algorithms [McG05, AG11, EKMS12, AG18, Tir18, GKMS19, ALT21, FMU22, AJJ+22, A23],
- random-order streams [KMM12, Kon18, ABB+19, GKMS19, FHM+20, Ber20, AB21, AS23],
- dynamic streams [Kon15, CCHM15, AKLY16, CCE+16, AKL17, DK20, AS22],
- weighted or submodular matchings [FKM+05, CS14, CK14, CGQ15, PS17, BDL21, LW21],
- matching size estimation [KKS14, EHL+15, BS15, MV16, CJMM17, MV18, AKL17, KMNT20, AKSY20, AN21, AS23],
- and, exact algorithms [FKM+05, GO13, AR20, LSZ20, CKP+21, AJJ+22].

In this paper, we focus on proving **multi-pass lower bounds for $(1-\varepsilon)$-approximation** of the maximum matching problem via semi-streaming algorithms, primarily for the regime of **small constant** $\varepsilon > 0$ independent of size of the graph.

The question of understanding the approximation ratio achievable by multi-pass semi-streaming algorithms for matchings was posed by [FKM+05] alongside the introduction of the semi-streaming model itself. Moreover, [FKM+05] also gave a $(2/3-\varepsilon)$-approximation algorithm for this problem in $O(1/\varepsilon)$ passes, which was soon after improved by [McG05] to a $(1-\varepsilon)$-approximation in $(1/\varepsilon)^{O(1/\varepsilon)}$ passes. A long line of work since then [AG11, KMM12, Kap13, EKMS12, KT17, AG18, Kon18, Tir18, ALT21, FMU22, AJJ+22] has culminated in semi-streaming algorithms with $\text{poly}(1/\varepsilon)$ passes for general graphs [FMU22] and $O(1/\varepsilon^2)$ passes for bipartite graphs [ALT21] (there are also algorithms with pass-complexity with better dependence on ε at the cost of mild dependence on n, namely, $O(\log n/\varepsilon)$ passes in [AG18, AJJ+22, A23] or for finding perfect matchings in $n^{3/4+o(1)}$ passes [AJJ+22]; see also [LSZ20]).

The lower bound front however has seen much less progress with only a handful of results known for single-pass algorithms [GKK12, Kap13, AKL17, Kap21] and very recently two-pass algorithms [KN21, A22]. But no lower bounds beyond two-pass algorithms are known for constant factor approximation of matchings in the semi-streaming model (lower bounds for computing exact maximum matchings up to almost $\Omega(\log n)$ passes are proven in [GO13]; see also [AR20, CKP+21], but these lower bounds cannot apply to $\varepsilon > n^{-o(1)}$, and we shall discuss them later in more details). It is worth noting that in the much more restricted setting of $\text{polylog}(n)$-space algorithms, [AN21], building on [AKSY20], proved an $\Omega(1/\varepsilon)$-pass lower bound for estimating the matching size.

1.1 Our Contribution

We present a new lower bound for multi-pass semi-streaming algorithms for the maximum matching problem. The lower bound is parameterized by the density of *Ruzsa-Szemerédi* (RS) graphs [RS78], namely, graphs whose edges can be partitioned into induced matchings of size $\Theta(n)$ (see Section 4.2). Let $\beta_{\text{RS}} \in (0,1)$ denote the largest parameter such that there exist n-vertex RS graphs with $\Omega(n^{\beta_{\text{RS}}})$ edge-disjoint induced matchings of size $\Theta(n)$. We prove the following result in this paper.

Result 1 (Formalized in Theorem 1)**.** *Any (possibly randomized) semi-streaming algorithm for $(1-\varepsilon)$-approximation of even the size of maximum matchings requires*

$$\Omega\Big(\frac{\log(1/\varepsilon)}{\log(1/\beta_{\text{RS}})}\Big)$$

passes over the stream. The lower bound holds for the entire range of $\varepsilon \in [n^{-\Theta(\beta_{\text{RS}})}, \Theta(\beta_{\text{RS}})]$.

To put this result in more context, we shall note that currently, it is only known that

$$\Omega\left(\frac{1}{\log\log n}\right) \underset{[\text{FLN}^+02]}{\leqslant} \beta_{\text{RS}} \underset{[\text{FHS17}]}{\leqslant} 1 - \Theta\left(\frac{\log^* n}{\log n}\right),$$

and closing this gap appears to be a challenging question in combinatorics (see, e.g. [Gow01, FHS17, CF13]). Thus, Result 1 can be interpreted as an $\Omega(\log(1/\varepsilon))$ lower bound on pass-complexity of $(1-\varepsilon)$-approximation of matchings in the semi-streaming model in two different ways:

- (i) A *conditional* lower bound, under the plausible hypothesis that $\beta_{\text{RS}} = \Omega(1)$. It is known how to construct RS graphs with induced matchings of size $n^{1-o(1)}$ that have $\binom{n}{2} - o(n^2)$ edges [AMS12], but the regime of $\Theta(n)$-size induced matchings is wide open.
- (ii) A *barrier result*; obtaining such algorithms requires reducing β_{RS} from $1-o(1)$ to $o(1)$ which seems beyond the reach of current techniques.

Let us now compare this result with some prior work.

A line of work closely related to ours is lower bounds for constant-approximation of matchings in one pass [GKK12, Kap13, AKL17, Kap21] or two passes [A22]. Specifically, [Kap21] rules out single-pass semi-streaming algorithms for finding (0.59)-approximate matchings (see also [GKK12, Kap13]). And, [AKL17] and [A22] rule out semi-streaming algorithms for approximating *size* of maximum matchings to within a $(1-\varepsilon_0)$ factor for some $\varepsilon_0 > 0$, in one and two passes[1], respectively (these two lower bounds, similar to ours, rely on the hypothesis that $\beta_{\text{RS}} = \Omega(1)$).

Another line of closely related work are lower bounds for computing perfect or nearly-perfect matchings in multiple passes [GO13, AR20, CKP+21], which culminated in the $\Omega(\sqrt{\log n})$ pass lower bound of [CKP+21] even for algorithms with $n^{2-o(1)}$ space (an almost $\Omega(\log n)$ pass lower bound for semi-streaming algorithms was already known by [GO13]). The lower bound of [CKP+21] can be further interpreted for $(1-\varepsilon)$-approximate matching algorithms as follows:

$$\begin{aligned}
&\Omega\Big(\frac{\log(1/\varepsilon)}{\sqrt{\log n}}\Big) \text{ pass lower bound when } \varepsilon \leqslant 2^{-\Theta(\sqrt{\log n})} \text{ for } n^{2-o(1)}\text{-space algorithms;}\\
&\Omega\Big(\frac{\log(1/\varepsilon)}{\log\log n}\Big) \text{ pass lower bound when } \varepsilon \leqslant (\log n)^{-\Theta(1)} \text{ for } \widetilde{O}(n)\text{-space algorithms.}
\end{aligned} \tag{1}$$

[1] See also [KN21] that give a two-pass lower bound for a restricted family of algorithms that only compute a greedy matching in their first pass but then can be arbitrary in their second pass.

Yet, these lower bounds, even under the strongest assumption of $\beta_{\text{RS}} = 1 - o(1)$ do not imply any non-trivial bounds for constant-factor approximation algorithms.

Before moving on from this section, we mention some important remarks about our result.

Constant-factor approximation. Our Result 1 is the *first* lower bound on pass-approximation tradeoffs for semi-streaming matching algorithms that applies to constant-factor approximations. The fact that the approximation can be a constant is critical here as we elaborate on below.

Firstly, in contrast to possibly some other models, in the semi-streaming model, the most interesting regime for $(1-\varepsilon)$-approximation is for constant $\varepsilon > 0$ (see, e.g. [FKM+05, McG05, Tir18, GKMS19, FMU22]). One key reason, among others, is that the cost of each additional pass over the stream is non-trivially high and thus algorithms that need super-constant number of passes over the stream (a consequence of sub-constant ε) are prohibitively costly already.

Secondly, many streaming matching algorithms have rather cavalier space-dependence on ε (as space is typically much less costly compared to passes), even exponential-in-ε, e.g., in [McG05, GKMS19, BDL21] (although see [ALT21, AJJ+22] for some exceptions with no space-dependence on ε at all). Yet, the lower bounds of the type obtained by [CKP+21] that require ε to be at most $(\log n)^{-\Theta(1)}$ (even assuming $\beta_{\text{RS}} = \Omega(1)$) cannot provide any meaningful guarantees for these algorithms, as the space of such algorithms for such small ε already become more than size of the input.[2] This however is not an issue for our lower bounds for constant-approximation algorithms.

Role of RS graphs. Starting from the work of [GKK12], all previous single- and multi-pass lower bounds for semi-streaming matching problem in [GKK12, Kap13, AKL17, AR20, Kap21, CKP+21, KN21, A22] are based on RS graphs—the only exception is the lower bound of [GO13] for finding perfect matchings (which is improved upon in [AR20, CKP+21] using RS graphs).

Our Result 1 is also based on RS graphs and relies on the hypothesis that $\beta_{\text{RS}} = \Omega(1)$ in order to be applicable to constant-factor approximation algorithms. While not all prior lower bounds rely on this hypothesis, assuming it also is not uncommon (see, e.g. [AKL17, A22]). Indeed, currently, a $(1-\varepsilon)$-approximation lower bound for estimating size of maximum matching that does not rely on this hypothesis is not known even for single-pass algorithms. Similarly, the space lower bound in Result 1 is in fact $n^{1+\Omega(1)}$; again, such bounds are not known even for finding edges of a $(1-\varepsilon)$-approximate matching in a single pass without relying on the $\beta_{\text{RS}} = \Omega(1)$ hypothesis. This in fact may not be a coincidence: a very recent work of [ABKL23] has provided evidence that at least qualitatively, relying on such hypotheses might be necessary. They use RS graph bounds algorithmically instead and show that if $\beta_{\text{RS}} = o(1)$, then one can find a $(1-\varepsilon)$-approximate matching already in a single pass in much better than quadratic space[3].

All in all, while we find the problem of proving (even single-pass) streaming matching lower bounds without relying on RS graphs, or even better yet, improving bounds on the density of RS graphs, quite fascinating open questions, we believe those questions are orthogonal to our research direction on multi-pass lower bounds.

Finally, we mention the current lower bound on β_{RS} due to [FLN+02, GKK12] combined with

[2] To give a concrete example, the state-of-the-art lower bounds before our paper left open the possibility of a $(1-\varepsilon)$-approximation algorithm in 3 passes and $(1/\varepsilon)^{O(1/\varepsilon)} \cdot \widetilde{O}(n)$ space. Obtaining such algorithms would have been a huge breakthrough and quite interesting. Our Result 1 however now rules out such an algorithm (conditionally) even in any $o(\log(1/\varepsilon))$ passes (or alternatively, identify a challenging barrier toward obtaining such algorithms).

[3] Quantitatively however, there is still a large gap between upper bounds of [ABKL23] even if $\beta_{\text{RS}} = o(1)$, and our bounds or those of [AKL17, A22] even if $\beta = 1 - o(1)$. Yet, this still suggests that the complexity of matching problem in graph streams is very closely tied to the density of RS graphs from both upper and lower bound fronts.

our Result 1 leads the following **unconditional result** for $(1-\varepsilon)$-approximation of matching size:

$$\Omega(\frac{\log(1/\varepsilon)}{\log\log\log n}) \text{ pass lower bound when } \varepsilon \leqslant (\log\log n)^{-\Theta(1)} \text{ for } \widetilde{O}(n)\text{-space algorithms,} \quad (2)$$

which exponentially improves the range of ε (and the denominator) compared to [CKP+21] in Eq (1).

Beyond $(\log(1/\varepsilon))$ **passes.** The pass lower bound in Result 1 (for $\beta_{\text{RS}} = \Omega(1)$) appears to hit the same standard barrier of proving super-logarithmic lower bounds for most graph streaming problems including reachability, shortest path, and perfect matching [GO13, AR20, CGMV20, CKP+21] (see [ACK19] for an in-depth discussion on this topic). Even for the seemingly algorithmically harder problem of finding a perfect matching, $\varepsilon = n^{-1}$, or nearly-perfect, $\varepsilon = n^{-\Omega(1)}$, the best lower bounds are only $\Omega(\log n) = \Omega(\log(1/\varepsilon))$ passes [GO13, CKP+21] (in contrast, the best known upper bounds for perfect matchings are $n^{3/4+o(1)}$ passes [AJJ+22]). Thus, going beyond $(\log(1/\varepsilon))$ passes seems to require fundamentally new techniques and the first step would be improving perfect matching lower bounds beyond $(\log n)$ passes, which is another fascinating open question.

1.2 Our Techniques

We follow the *set hiding* approach of [AR20, CKP+21, A22] and an elegant recursive framework of [CKP+21] that achieves *permutation hiding* (a primitive entirely missing from [AR20, A22] and seemingly crucial for proving more than two-pass lower bounds). At a high level, p-pass permutation hiding graphs hide a unique permutation of vertex-disjoint augmenting paths—necessary for finding large enough matchings—, in a way that a semi-streaming algorithm cannot find this permutation in p passes (see Section 2.2); a set hiding graph roughly corresponds to only hiding the endpoints of these paths. [CKP+21] shows a way of constructing p-pass set hiding graphs from $(p-1)$-pass permutation hiding graphs (that can be made efficient), and constructing p-pass permutation hiding graphs from p-pass set hiding graphs rather inefficiently by blowing up the number of vertices by a $\Theta(\log n)$ factor. This results in having to reduce ε to $\varepsilon/\Theta(\log n)$ for each application of this idea in each pass, leading to a lower bound of $\Omega(\log(1/\varepsilon)/\log\log n)$ passes eventually.

In a nutshell, we present a novel approach for directly constructing permutation hiding graphs, without cycling through set hiding ones first and thus avoiding the $\Theta(\log n)$ overhead of [CKP+21] in vertices and the approximation factor. Hence, we only need to reduce ε to some $\Theta(\varepsilon)$ for each pass, leading to our $\Omega(\log(1/\varepsilon))$ lower bound. Conceptually, the technical novelty of our paper can be summarized as working with permutations *directly* both in the construction and in the analysis.

We first give a novel combinatorial approach for hiding permutations directly inside RS graphs, instead of only using them for hiding Boolean strings and applying Boolean operators on top of RS graphs as was done previously (e.g., $\wedge$- or $\vee$- operators of [CKP+21]). This step uses various ideas developed in [GKK12, AKL17, AR20, AB21, AS23, A22] (see [A22, Section 1.2] for an overview of these techniques) that can then be combined with the general framework of [CKP+21] in a non-black-box way, for instance, by replacing sorting network ideas of [CKP+21] with k-sorter networks in [PP89, Chv92] with lower depth (and various technical changes in the analysis).

The second and the main technical ingredient of our paper is to introduce and analyze a "permutation variant" of the *Boolean Hidden (Hyper)Matching (BHH)* communication problem of [GKK+07, VY11]. The BHH problem, alongside its proof ideas, has been a key ingredient of streaming matching lower bounds, among many others, in recent years [AKSY20, CKP+21, AN21, KMT+22, AS23, A22] (see [AKSY20, Appendix B] for an overview). Roughly speaking, our problem (see Section 2.1), replaces the hidden Boolean strings in BHH and their XOR operator with permutations and the concatenation operator. Its analysis then boils down to proving a hardness

amplification result for the concatenation of permutations, quite similar in spirit to XOR Lemmas for Boolean strings. While these XOR Lemmas are primarily proven using Fourier analysis on Boolean hypercube (see, e.g., [GKK+07, VY11, KKS15, KMT+22, AS23, A22]), and in particular the KKL inequality [KKL88], we prove our results by working with basic tools from representation theory and Fourier analysis on symmetric groups, combined with detailed information-theoretic arguments, including a recent KL-divergence vs ℓ_1/ℓ_2-distance inequality of [CK18].

2 Main Result

We present our main theorem in this section that formalizes Result 1 from the introduction, plus the key ingredients we use to prove it.

We define a bipartite graph $G_{\text{rs}} = (L_{\text{rs}}, R_{\text{rs}}, E_{\text{rs}})$ to be a $(2n^{\text{rs}})$-vertex bipartite (r,t)-RS graph if its edges can be partitioned into t induced matchings $M_1^{\text{rs}}, \ldots, M_t^{\text{rs}}$ each of size r; here, an induced matching means that there are no other edges between the endpoints of the matching. See Section 4.2 for more details on RS graphs.

Theorem 1 (Formalization of Result 1)**.** *Suppose that for infinitely many choices of $n^{\text{rs}} \geqslant 1$, there exists $(2n^{\text{rs}})$-vertex bipartite (r,t)-RS graphs with $r = \alpha \cdot n^{\text{rs}}$ and $t = (n^{\text{rs}})^{\beta}$ for some fixed parameters $\alpha, \beta \in (0,1)$; the parameters α, β can depend on n^{rs}.*

Then, there exists an $\varepsilon_0 = \varepsilon_0(\alpha,\beta)$ such that the following is true. For any $0 < \varepsilon < \varepsilon_0$, any streaming algorithm that uses $o(\varepsilon^2 \cdot n^{1+\beta/2})$ space on n-vertex bipartite graphs and can determine with constant probability whether the input graph has a perfect matching or its maximum matchings have size at most $(1-\varepsilon) \cdot n/2$ requires

$$\Omega\Big(\frac{\log(1/\varepsilon)}{\log(1/\alpha\beta)}\Big)$$

passes over the stream.

Result 1 then follows from Theorem 1 by setting $\alpha = \Theta(1)$ and using the bound $\varepsilon > n^{-\beta/6}$ to obtain a space lower bound of $\Omega(n^{1+\beta/6})$ for $o(\log(1/\varepsilon)/\log(1/\beta))$ pass algorithms by Theorem 1. This is because any $(1-\varepsilon)$-approximation of size of maximum matchings distinguishes between the two families of the graphs in the theorem. Finally, given that we know by [FLN+02, GKK12] that $\beta = \Omega(1/\log\log(n))$, the space bound of $o(n^{1+\beta/6})$ will always rule out semi-streaming algorithms.

We now go over the two main ingredients in the proof of this theorem, and state our main results for them. In the next section, we present a proof outline of each of these ingredients, plus that of Theorem 1. The rest of the paper is then dedicated to formalizing these proof outlines.

2.1 Ingredient I: (Multi) Hidden Permutation Hypermatching

A key to our lower bound constructions is a problem in spirit of the Boolean Hidden Hypermatching (BHH) problem of [VY11] (itself based on [GKK+07]) that we introduce in this paper. The definition of the problem is rather lengthy and can be daunting at first, so we build our way toward it by considering BHH first, and then move from there.

The Boolean Hidden Hypermatching (BHH) problem

The BHH problem can be phrased as follows (this is slightly different from the presentation in [VY11] but is equivalent to the original problem). We have Alice with a string $x \in \{0,1\}^{r \times k}$ and Bob who has a hypermatching M over $[r]^k$ with size $r/2$ plus a string $w \in \{0,1\}^{r/2}$. The players

are promised that the parity of x on hyperedges of M, i.e.,

$$M \cdot x = (\oplus_{i=1}^{k} x_{M_{1,i},i} \ , \ \oplus_{i=1}^{k} x_{M_{2,i},i} \ , \ \cdots \ , \ \oplus_{i=1}^{k} x_{M_{r/2,i},i}) \in \{0,1\}^{r/2}$$

is either equal to w or $\bar{w}$. The goal is for Alice to send a single message to Bob and Bob outputs which case the input belongs to. It is known that $\Theta(r^{1-1/k})$ communication is necessary and sufficient for solving BHH with constant probability [VY11].

A natural variant of BHH (defined as a direct-sum version of BHH) is also used in [CKP+21] as one of the main building blocks for constructing their permutation hiding graphs. Roughly speaking, in that problem, Alice is given several different strings x and Bob's input additionally identifies which string to compute the parities of hyperedges of M over.

Nevertheless, the Boolean nature of this problem is too restrictive for the purpose of our constructions (and in fact, this Boolean nature is the key bottleneck in the construction of [CKP+21]). Thus, for our purpose, we define a "permutation variant" of this problem.

The Hidden Permutation Hypermatching (HPH) Problem

We define the **Hidden Permutation Hypermatching (HPH)** problem as follows. Let $b \geqslant 1$ and S_b be the set of permutations over $[b]$. We have a permutation matrix $\Sigma \in (S_b)^{r \times k}$ and a hypermatching M over $[r]^k$ with size $r/2$, plus a permutation vector $\Gamma \in (S_b)^{r/2}$. We are promised:

$$\Gamma^* = \left(\circ_{i=1}^{k} \sigma_{M_{1,i},i} \ , \ \circ_{i=1}^{k} \sigma_{M_{2,i},i} \ , \ \cdots \ , \ , \circ_{i=1}^{k} \sigma_{M_{r/2,i},i} \right) \in (S_b)^{r/2}$$

is such that $\Gamma^* \circ \Gamma$ is either equal to one of the two fixed known permutation vectors Γ_{Yes} or Γ_{No}; here, $\circ_{i=1}^{k}(\cdot)$ concatenates the given k permutations together. The goal is to distinguish which case the input belongs to. See Figure 1 for an illustration.

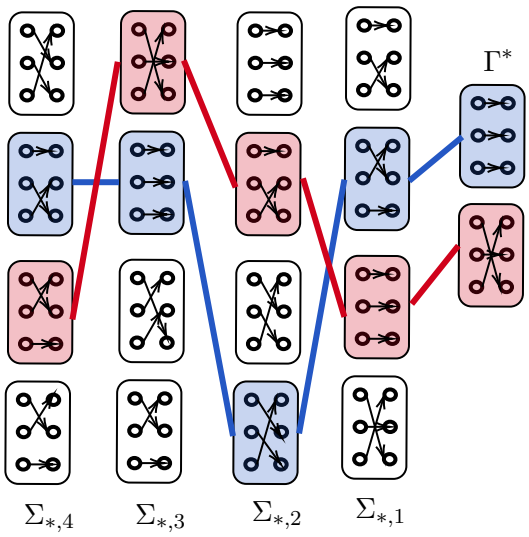


Figure 1: An illustration the HPH problem for $r = 4$, $k = 4$ and $b = 3$. We have $r/2 = 2$ hypermatching edges corresponding to the thick edges (red and blue, respectively, for each hyperedge).

One can see that this problem is equivalent to BHH whenever $b = 2$: we can simply interpret the identity permutation in S_2 as a 0-bit and the cross permutation as a 1-bit; the concatenation operator in this case will then become XOR naturally and the problem will be identical to BHH.

Nonetheless, once we move on to larger values of $b > 2$, this problem becomes much "richer" than BHH as it can encode $b!$ different "states" per each entry and the concatenation operator becomes quite different than XOR. Indeed, there are already generalizations of BHH by replacing Boolean domain with finite fields $\mathbb{F}_b$ for $b > 2$, and the XOR operator with addition in this field [GT19]; yet, even those generalizations only correspond to very limited types of permutations and concatenations, and are strict special cases of HPH (which also do not work for our constructions as they can only hold b "states" per entry as opposed to $b!$).

There is however an important subtlety in the definition of this problem in our paper that we need to clarify. While we could have turned HPH into a communication game, exactly as in BHH, by giving Σ to Alice and M, Γ to Bob, this is *not* what we do actually. Instead, we partition Σ column-wise between k different players $Q^{(1)}, \ldots, Q^{(k)}$ by providing each player $Q^{(i)}$ for $i \in [k]$ with the permutation vector $\Sigma_{*,i} \in (S_b)^r$, the i-th column of Σ, and provide (M, Γ) to a referee (identical to Bob). The communication pattern is also different in that, first, $Q^{(1)}, \ldots, Q^{(k)}$ can talk with each other, with back and forth communication, using a shared blackboard visible to all parties. Then, at the end of their communication, the referee can check the content of the blackboard and output the answer (think of the final state of the blackboard as the message of players to the referee). We shall discuss the technical reasons behind this change in our proof outline in Section 3.1.

The Multi Hidden Permutation Hypermatching (Multi-HPH) Problem

We are now ready to present the full version of the problem we study, which can be seen as a certain direct-sum variant of the HPH (similar in spirit to the way BHH is generalized in [CKP+21]). Roughly speaking, in this problem, there are t instances of HPH and the referee additionally chooses which instance of the HPH they should all solve, without the other players knowing this information at the time of their communication.

We do caution the reader that since this problem involves "tensorizing" the inputs in HPH (e.g., Σ becoming a 3-dimensional tensor and each player receiving a permutation matrix), the indices stated below may not directly map to the ones stated earlier for HPH (which is the $t = 1$ case).

Problem 1 (**Multi-Hidden-Permutation-Hypermatching**). *For any integers $r, t, b, k \geqslant 1$, **Multi-HPH**$_{r,t,b,k}$ is a distributional $(k+1)$-communication game, consisting of k players plus a referee, defined as:*

*(i) Let $\Gamma_{\text{Yes}}, \Gamma_{\text{No}} \in (S_b)^{r/2}$ be two arbitrary tuples of permutations known to all $(k+1)$ parties. We refer to Γ_{Yes} and Γ_{No} as the **target tuples**.*

(ii) We have k players $Q^{(1)}, \ldots, Q^{(k)}$ and for all $i \in [k]$ the i-th player is given a permutation matrix $\Sigma^{(i)} \in (S_b)^{t \times r}$ chosen independently and uniformly at random.

(iii) The referee receives k indices $L := (\ell_1, \ldots, \ell_k)$ each picked uniformly and independently from $[t]$ and a hypermatching $M \subseteq [r]^k$ with $r/2$ hyperedges picked uniformly.

Additionally, let the permutation vector $\Gamma^* = \left(\gamma_1^\star, \gamma_2^\star, \ldots, \gamma_{r/2}^\star\right) \in (S_b)^{r/2}$ be defined as,

$$\forall a \in [r/2] \;\; \gamma_a^\star := \sigma^{(1)}_{\ell_1, M_{a,1}} \circ \cdots \circ \sigma^{(k)}_{\ell_k, M_{a,k}},$$

where $M_{a,i}$ for $a \in [r/2]$ and $i \in [k]$ refers to the i-th vertex of the a-th hyperedge in M.

The referee is also given another permutation vector $\Gamma = (\gamma_1, \gamma_2, \ldots, \gamma_{r/2})$ sampled from $(S_b)^{r/2}$ conditioned on $\Gamma^* \circ \Gamma$ being equal to either Γ_{Yes} or Γ_{No} (here, the $\circ$ operator is used to concatenate each permutation in the vectors individually).

The players $Q^{(1)}, \ldots, Q^{(k)}$ can communicate with each other by writing on a shared board visible to all parties with possible back and forth and in no fixed order (the players' messages are functions of their inputs and the board). At the end of the players' communication, the referee can use all these messages plus its input and outputs whether $\Gamma^* \circ \Gamma$ is Γ_{Yes} or Γ_{No}.

The following theorem on the communication complexity of **Multi-HPH** involves the bulk of our technical efforts in this paper.

Theorem 2. *For any integers $t \geqslant 1$, $b \geqslant 2$, and sufficiently large $r, k \geqslant 1$, any protocol for $\textbf{Multi-HPH}_{r,t,b,k}$, for any pairs of target tuples, with at most s bits of total communication for s satisfying*

$$k \cdot \log(r \cdot t) \leqslant s \leqslant 10^{-3} \cdot (r \cdot t)$$

can only succeed with probability at most

$$\frac{1}{2} + r \cdot O\Big(\frac{s}{r \cdot t}\Big)^{k/32}.$$

In our lower bounds, we use this theorem with parameters $k = \Theta(1/\beta)$, and $s = r \cdot \sqrt{t}$ (where r and $t \approx r^{\beta}$ will be determined by the underlying (r, t)-RS graph we use). This allows us to prove a lower bound for protocols that can solve **Multi-HPH** with advantage as as small as $1/\text{poly}(r)$ over random guessing, which will be crucial for our constructions.

2.2 Ingredient II: Permutation Hiding Graphs

We now present the definition of permutation hiding graphs, introduced by [CKP+21], that are also a key building block in our lower bound. To do this, we need some notation.

We call a directed acyclic graph $G = (V, E)$ a **layered** graph if its vertices can be partitioned into sets $V^1, \ldots, V^d$ for some $d \geqslant 1$, such that any edge of G is directed from some V^i to V^{i+1} for some $i \in [d-1]$; we further define $\textsc{First}G := V^1$ and $\textsc{Last}G := V^d$.

We define a **layered graph** as $G = (V, E)$ as a directed acyclic graph whose vertex set V can be partitioned into d different sets $V_1, V_2, \ldots, V_d$ where any edge (u, v) is such that $u \in L_i$ and $v \in L_{i+1}$ for some $i \in [d-1]$. We use $\textsc{First}(G)$ and $\textsc{Last}(G)$ to denote the sets V_1 (the first layer of G) and V_ℓ (the last layer of G) respectively. We index each layer V_i by the set $[|V_i|]$ for each $i \in [d]$. We use $\textsc{First}_{[i]}(G)$ to denote the first i vertices of $\textsc{First}(G)$ indexed by the set $[i]$ for any $i \in [|\textsc{First}(G)|]$ and similarly for $\textsc{Last}_{[j]}(G)$ for $j \in [|\textsc{Last}(G)|]$.

We will use layered graphs to represent permutations over $[m]$, denoted by S_m, as follows.

Definition 2.1 (Permutation Graph)**.** For any integer $m \geqslant 1$, a layered graph $G = (V, E)$ is said to be a **permutation graph** for $\sigma \in S_m$ if $|\textsc{First}(G)|, |\textsc{Last}(G)| \geqslant m$ and there is a path from $i \in \textsc{First}_{[m]}(G)$ to $j \in \textsc{Last}_{[m]}(G)$ if and only if $\sigma(i) = j$ for each $i, j \in [m]$.

We use $\mathcal{D}_m$ to denote the set of all permutation graphs on S_m. We are now ready to define the (distribution of) permutation hiding graphs. Our definition is a slight rephrasing of the one in [CKP+21] and we claim no novelty here.

Definition 2.2 (Permutation Hiding Graphs; c.f. [CKP+21])**.** For integers $m, n, p, s \geqslant 1$ and real $\delta \in (0,1)$, we define a **permutation hiding generator** $\mathcal{G} = \mathcal{G}(m, n, p, s, \delta)$ as any family of distributions $\mathcal{G} : S_m \to \mathcal{D}_m$ on permutation graphs satisfying the following two properties:

(*i*) For any $\sigma \in S_m$, any permutation graph G in the support of $\mathcal{G}(\sigma)$ has n vertices.

(*ii*) For any $\sigma_1, \sigma_2 \in S_m$, the distribution of graphs $\mathcal{G}(\sigma_1)$ and $\mathcal{G}(\sigma_2)$ are δ-indistinguishable for any p-pass s-space streaming algorithm.

[CKP+21] presented a permutation hiding generator with the following parameters for any integers $m, p \geqslant 1$ in terms of the parameters α and β of RS graphs:

$$n := (\frac{\log n}{\alpha})^{\Theta(p)} \cdot \Theta(m); \qquad \text{(number of vertices)}$$
$$s := o(m^{1+\beta}); \qquad \text{(space of streaming algorithm)}$$
$$\delta := 1/\text{poly}(n). \qquad \text{(probability of success of the algorithm)}$$

In particular, notice that the number of vertices in the graph grows by a factor of $\log(n)$ per pass even when both α and β are constant. As we shall see later in this section, the ratio of m/n governs the approximation ratio of the algorithms for the maximum matching problem. As such, the ratio in this construction is too small to provide lower bounds for constant-factor approximation streaming algorithms, no matter the space of the algorithm.

We present an alternative construction of permutation hiding graphs in this paper, which is more suited for proving streaming lower bounds for maximum matching.

Theorem 3. *For any integers $m, p \geqslant 1$, there is a permutation hiding generator $\mathcal{G} = \mathcal{G}(m, n, p, s, \delta)$ with the following parameters:*

$$n := \Theta(\frac{1}{\alpha \cdot \beta^2})^{p} \cdot \Theta(m); \qquad \text{(number of vertices)}$$
$$s := o(m^{1+\beta/2}); \qquad \text{(space of streaming algorithm)}$$
$$\delta := (p/\beta)^{\Theta(1/\beta)} \cdot \Theta(1/\beta)^{2p} \cdot 1/\text{poly}(m). \qquad \text{(probability of success of the algorithm)}$$

Thus, we obtain a different tradeoff than [CKP+21] on the parameters of the permutation hiding generator. In particular, now, the ratio of n/m is only $2^{\Theta(p)}$ for constant values of α, β, which as we shall see soon, is sufficient to obtain our $\Omega(\log(1/\varepsilon))$-pass lower bound.

Our proof of Theorem 3 considerably deviates from that of [CKP+21] both in terms of the combinatorial construction of the permutation hiding graphs and even more so in the information-theoretic analysis of their properties. Since the majority of these changes are already apparent even for single-pass algorithms, in Section 6 we first present the construction for single-pass algorithms separately as a warm-up to our main construction. Section 7 then contains the construction and analysis for multi-pass algorithms using the same type of inductive argument as in [CKP+21] by replacing their induction step with our approach in Section 6 for single-pass algorithms.

3 Proof Outline

We present a proof outline of Theorem 1 and its ingredients in this section. We emphasize that this section oversimplifies many details and the discussions will be informal for the sake of intuition.

We will start with the two key ingredients of our main theorem, namely, Theorems 2 and 3, and then at the end, show how they can easily imply the main theorem as well.

3.1 Proof Outline of Theorem 2: (Multi) Hidden Permutation Hypermatching

We start with the proof outline of Theorem 2. The formal proof is presented in Section 5.

3.1.1 From XOR Lemmas to a "Concatenation Lemma"

Let us focus on the HPH problem wherein the input is a permutation matrix $\Sigma \in (S_b)^{r \times k}$ given to k players $Q^{(1)}, \ldots, Q^{(k)}$, and a hypermatching M, and permutation vector $\Gamma \in (S_b)^{r/2}$ given to the referee. The goal is to determine whether for the permutation vector Γ^* obtained via concatenating permutations on indices of M, $\Gamma^* \circ \Gamma$ is equal to a fixed permutation vector Γ_{Yes} or another vector Γ_{No}. The communication pattern is as specified in Problem 1.

Our starting point is similar to that of [GKK+07, VY11] by breaking the correlation in the input instance (either all permutations in $\Gamma^* \circ \Gamma$ are consistent with Γ_{Yes} or all are consistent with Γ_{No}). Roughly speaking, this corresponds to showing that if we only have a single random hyperedge $e = (v_1, \ldots, v_k) \in [r]^k$, then, the distribution of

$$\gamma^* = \sigma_{v_1,1} \circ \cdots \circ \sigma_{v_k,k};$$

is $1/\text{poly}(r)$ close to the uniform distribution in the total variation distance. Having proven this, one can then essentially do a "union bound" and show that the distribution of the entire vector $\Gamma^* \circ \Gamma$ remains so close to uniform that the referee cannot distinguish whether it is consistent with Γ_{Yes} or Γ_{No} in this case. Formalizing this step is not immediate and requires a *hybrid argument* similar to those of [GKK+07, VY11] (see [AN21, Section 4.3] also for a general treatment) but we shall skip it in this discussion.

Concretely, our task is to prove the following inequality for any low communication protocol π for HPH, with a transcript Π between its k players $Q^{(1)}, \ldots, Q^{(k)}$:

$$\mathop{\mathbb{E}}_{(\Pi,e)} \| (\gamma^* \mid \Pi, e) - \mathcal{U}_{S_b} \|_{\text{tvd}} \leqslant 1/\text{poly}(r), \tag{3}$$

where $\mathcal{U}_{S_b}$ is the uniform distribution over S_b and $\| \cdot \|_{\text{tvd}}$ is the total variation distance (TVD).

XOR Lemmas. By the equivalence between HPH and BHH for $b = 2$, proving the equivalent of Eq (3) in $b = 2$ case for BHH in all prior work [GKK+07, VY11, KKS15, CKP+21, KMT+22, AS23, A22] corresponds to proving an *XOR Lemma* for the *Index* communication problem: We have a string $x \in \{0,1\}^{r \times k}$ conditioned on a short message Π, and we are interested in $\oplus_{i=1}^{k} x_{v_i,i}$ for $(v_1, \ldots, v_k)$ chosen uniformly from $[r]^k$. At an intuitive level, these works rely on the following two statements: (i) since Π is a short message, each $x_{v_i,i}$ should be individually somewhat close to uniform distribution (by the standard lower bounds for the Index problem [Abl93, KNR95]), and (ii) since we are taking XOR of multiple close-to-uniform bits, the final outcome should be even exponentially-in-k closer to uniform[4]. Formalizing this intuition is done via different tools such as Fourier analysis on Boolean hypercube [GKK+07, VY11, KKS15, KMT+22, AS23, A22], discrepancy bounds [CKP+21] or a generic streaming XOR Lemma [AN21].

[4] This step is easy to see *had* the message Π was not correlating the values of $x_{v_i,i}$ across different $i \in [k]$: the bias of XOR of independent bits is equal to multiplication of their biases (see, e.g. [AN21, Proposition A.9]). Yet the message can indeed correlate these values and the main challenge in proving any XOR Lemma is to handle this. We refer the reader to Yao's XOR Lemma [Yao82] for the first example of such approaches in circuit complexity, and [AN21] for a streaming XOR Lemma and [Yu22] for a bounded-round communication XOR Lemma.

Our approach here is then to extend these XOR Lemmas to a **"Concatenation Lemma"** for Eq (3), which we describe in the next part.

3.1.2 A "Concatenation Lemma"

Following the previous discussion, our goal in proving Eq (3) is to show that: (i) since Π is a short message, each permutation $\sigma_{v_i,i}$ should be individually somewhat close to the uniform distribution (which still follows from a similar argument as for the Index problem [Abl93,KNR95]), and (ii) since we are taking concatenation of multiple close-to-uniform permutations, the final outcome should be even exponentially-in-k closer to uniform. This is what we consider a "concatenation lemma" in this paper, and it is where we start to deviate completely from prior approaches in [GKK+07, VY11, KKS15, CKP+21, KMT+22, AS23, A22] in this context.

A Conceptual Roadblock. Let us point out an important conceptual challenge in formalizing step (ii) of this plan. Let ν denote the uniform distribution over all *even* permutations on S_b, namely, permutations with an even number of inversions. We have that the TVD of ν from $\mathcal{U}_{S_b}$ is $1/2$, so, roughly speaking, ν is already "not-too-far" from the uniform distribution.

But now consider the distribution ν^* which is the k-fold concatenation of ν for some $k \geqslant 1$, meaning that to sample from ν^*, we sample k independent permutations from ν and concatenate them together. To be able to implement step (ii) of our plan, at the very least, we should have that in this purely independent case, TVD of ν^* from $\mathcal{U}_{S_b}$ exponentially drops, i.e., becomes $2^{-\Omega(k)}$. Alas, it is easy to see that ν^* is in fact the same as ν and thus we have *no* change in TVD at all! This is in stark contrast with the XOR and Boolean case where the drop in TVD, at least for purely independent inputs, is *always* happening.

3.1.3 Our Proof Strategy for the "Concatenation Lemma"

With the above example in mind, we are now ready to discuss our solution for proving the Concatenation Lemma and establishing Eq (3). For now, let us limit ourselves to *nice* protocols that do not correlate the outcome of different permutations with each other, meaning that we are still in this blissful case wherein the permutations $\sigma_{v_i,i}$ are independent even conditioned on Π. The first and easy step of the argument is to show that the distribution of each $\sigma_{v_i,i}$, for a random v_i, is close to uniform not only in TVD but also KL-divergence, namely,

$$\mathop{\mathbb{E}}_{\Pi,v_i}\left[\mathbb{D}(\sigma_{v_i,i} \mid \Pi \mid\mid \mathcal{U}_{S_b})\right] \leqslant r^{-1/2}. \tag{4}$$

This step is still not particularly different from the typical Index lower bounds and is an easy application of the chain rule of KL-divergence. Let us make one further simplifying assumption by taking Eq (4) to hold, not in expectation, but rather simultaneously for all coordinates of a fixed $(v_1, \ldots, v_k)$ at the same time.

At this point, one could apply Pinsker's inequality (Fact A.12) to relate the KL-divergence bound in Eq (4) to a bound on TVD, but then we may end up in the situation shown in the roadblock, hence not allowing for further decay in the distance through concatenation.

Another approach, taken for instance in [AKSY20], is to relate Eq (4) to an ℓ_2-distance of the distributions (instead of TVD which is half the ℓ_1-distance). But then, the best bound one can prove on ℓ_2-distance will also be $r^{-\Theta(1)}$, which again would not suffice for our purpose (using a "smooth" version of the example in the roadblock; see Appendix C).

Both above cases suggest that we may need a more nuanced understanding of the distribution of $\sigma_{v_i,i} \mid \Pi$. To do so, we consider a combination of TVD and ℓ_2-distances in the following way. We

apply the "KL-vs-ℓ_1/ℓ_2-inequality" of [CK18] (Proposition A.13)—a generalization of the Pinsker's inequality—to decompose the support of the distribution of $\sigma_{v_i,i} \mid \Pi$ into two parts:

- A_i: A part that does *not* happen that frequently, meaning $\sigma_{v_i,i} \mid \Pi$ is only in A_i w.p. $r^{-1/4}$;
- B_i: A part that is *extremely close* to uniform distribution in ℓ_2-distance, meaning

$$\|(\sigma_{v_i,i} \mid \Pi, B_i) - \mathcal{U}_{S_b}\|_2 \leqslant \frac{r^{-\Theta(1)}}{b!}. \tag{5}$$

Putting these together, plus our simplifying assumption that $\sigma_{v_i,i}$'s are still independent conditioned on Π allows us to argue that with probability $1 - r^{-\Theta(k)}$, there are at least $\Theta(k)$ coordinates $i \in [k]$ that satisfy Eq (5).

This brings us to the last part of the argument. Having obtained $\Theta(k)$ coordinates that are extremely close to uniform, we use basic tools from representation theory and Fourier analysis on permutations to analyze the distribution of their concatenation. The Fourier basis here is a set of irreducible representation matrices (see Appendix B), and the convolution theorem allows us to relate Fourier coefficients of the concatenated permutation via multiplication of Fourier coefficients of each individual permutation. Finally, the distance to the uniform distribution can be bounded by Plancharel's inequality for this Fourier transform (Proposition B.5), similar to the standard analysis on the Boolean hypercube (see, e.g., [dW08]).

All in all, this step allows us to bound the TVD of the concatenation of the permutation—conditioned on the case of $\Theta(k)$ extremely-close to uniform indices in $[k]$ which happens with probability $1 - r^{-\Theta(k)}$—by another $r^{-\Theta(k)}$. Putting all these together then gives us the desired inequality in Eq (3) under all our earlier simplifying assumptions.

Removing simplifying assumptions. The above discussion oversimplified many details, chief among them, the main challenge that stems from the inputs becoming correlated through the transcript Π (which is the *key* challenge in proving XOR Lemmas as well). For the BHH problem and XOR Lemmas (for the Index problem) in the Boolean setting, a key tool to handle this is the *KKL inequality* of [KKL88] for the Fourier transform on Boolean hypercube, which as a corollary, almost immediately gives an XOR Lemma for the Index problem (see [dW08, Section 4.2]). For our permutation problem, however, we are not aware of any similar counterpart.

We handle the aforementioned challenge by replacing the role of a single player, Alice, in BHH with k separate players in HPH and use a detailed information-theoretic argument to analyze how much these players can correlate their inputs. Roughly speaking, this reduces the problem to proving that the inputs of players are only correlated through the message Π and not beyond that (a consequence of the rectangle property of communication protocol), and then making a *direct product* argument to show a single message Π, cannot, *simultaneously*, change the distribution of multiple coordinates.

3.1.4 From HPH to Multi-HPH: (Not) A Direct-Sum Result

To prove Theorem 2, we need a lower bound for the **Multi-HPH** problem which is stronger than the lower bound for HPH by a factor of t. Given that **Multi-HPH** is effectively a direct-sum version of HPH—we need to solve one *unknown* copy out of t given copies—it is natural to expect the complexity of the problem also increases by a factor t. Moreover, given various direct-sum results known using *information complexity* (see, e.g. [CSWY01, JRS03, BBCR10, BR11, BRWY13] and references therein), one might expect this step to be an easy corollary.

Unfortunately, this is in fact not the case, due to the crucial reason that we need a lower bound for protocols with an extremely small advantage of only $1/\text{poly}(r)$ over random guessing. In general, one should not expect a generic direct sum result to hold in this low-probability regime[5]. Moreover, these information complexity approaches typically fail on problems with a low probability of success, and in our case, they cannot be applied readily.[6]

Consequently, in our proof of Theorem 2, we directly work with the **Multi-HPH** problem, which means all the arguments stated in the previous part should be implemented for this "higher" dimensional problem. Nevertheless, most of these changes appear in the first part of the argument that reduces the original problem to proving Eq (3) and Eq (4) (or rather their equivalents for **Multi-HPH**), as well as the direct product arguments mentioned at the end of the last subsection for removing our simplifying assumptions. Thus, these changes do not fundamentally alter our previously stated plan in the lower bound and we postpone their details to Section 5.

3.2 Proof Outline of Theorem 3: Permutation Hiding Graphs

We now switch to the proof outline of Theorem 3. The formal proof is presented in Section 6 for single-pass algorithms and Section 7 for multi-pass ones.

Our proof primarily builds on [CKP+21] (for the general framework) and [A22] (for the construction of the graphs used in the framework). Both these papers have excellent overviews of their technical approach, [CKP+21, Section 2] and [A22, Section 1.2], and we refer the reader to those parts for further background as well as an overview of prior techniques.

As stated earlier, our main point of departure from the framework of [CKP+21] already appears for single-pass algorithms. So, in this overview also, we mostly focus on this case. For the simplicity of exposition, in the following, we assume the parameters α, β of RS graphs are both $\Theta(1)$ and ignore the dependence of our bounds on them.

3.2.1 Permutation Hiding for Single-Pass Algorithms

Recall the definition of permutation hiding graphs from Definition 2.2. Theorem 2 for **Multi-HPH** is our main tool for constructing these graphs, so let us see how we can turn an instance (Σ, L, M, Γ) of **Multi-HPH** into a permutation graph. We start with each component separately.

Encoded RS graphs and Σ

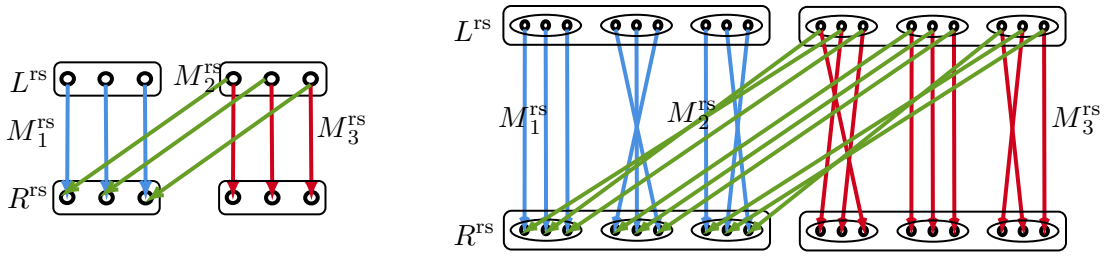
Recall that Σ consists of k permutation matrices $\Sigma^{(i)} \in (S_b)^{t \times r}$. Let $G_{\text{rs}} = (L_{\text{rs}}, R_{\text{rs}}, E_{\text{rs}})$ be a bipartite RS graph with n^{rs} vertices on each side and t induced matchings $M_1^{\text{rs}}, \ldots, M_t^{\text{rs}}$ each of size r (for the same parameters as the dimensions of $\Sigma^{(i)}$).

We "encode" $\Sigma^{(i)}$ in G_{rs} via the following "graph product" $G_{\text{rs}} \otimes \Sigma^{(i)}$ (strictly speaking, this is the product of the graph G_{rs} and the matrix $\Sigma^{(i)}$):

- Vertices of $G_{\text{rs}} \otimes \Sigma^{(i)}$ are the bipartition $L^{(i)} = L_{\text{rs}} \times [b]$ and $R^{(i)} = R_{\text{rs}} \times [b]$.
- Edges of $G_{\text{rs}} \otimes \Sigma^{(i)}$ are directed from $(u, x) \in L$ to $(v, y) \in R$ whenever (u, v) is an edge in E *and* $y = \sigma^{(i)}_{c_1, c_2}(x)$ where $(c_1, c_2) \in [t] \times [r]$ is chosen so that e is the c_2-th edge of the c_1-th induced matching $M^{\text{rs}}_{c_1}$.

[5] A short explanation is based on the equivalence of information complexity and direct sum result [BR11], plus the fact that information complexity is an "expected" term while communication complexity is a "worst case" measure.

[6] Concretely, a protocol that with probability $1/\text{poly}(r)$, communicates its input and otherwise is silent has an $O(1)$ information complexity (albeit large communication complexity) and can lead to the desired advantage of $1/\text{poly}(r)$ trivially; thus information complexity of HPH in such a low-probability-of-success regime is simply $O(1)$.



(a) A $(3,3)$-RS graph G_{rs}.

(b) $G_{\text{rs}} \otimes \Sigma$ for some unspecified 3×3 permutation matrix Σ.

Figure 2: An illustration of RS graphs and our "graph product".

In words, we "stretch" each vertex of G_{rs} to become b separate vertices, and replace the edge $e = (u, v) \in E_{\text{rs}}$ by a matching of size b between vertices $(u, *)$ and $(v, *)$ in the product; the choice of this matching is then determined by the entry of $\Sigma^{(i)}$ that "corresponds" to this edge, where in this correspondence we interpret rows of $\Sigma^{(i)}$ as the induced matchings in G_{rs}, and the columns as the edges of these matchings. See Figure 2 for an illustration.

It is not hard to verify that $G_{\text{rs}} \otimes \Sigma^{(i)}$ is itself an RS graph with t induced matchings of size $r \cdot b$ on $n^{\text{rs}} \cdot b$ vertices. This means that the product has "sparsified" the original RS graph relatively, but we shall ensure that b is sufficiently small, such that this product graph is also sufficiently dense still for the purpose of our lower bound.

We can then encode the entirety of Σ into k vertex-disjoint RS graphs $G_{\text{rs}} \otimes \Sigma^{(i)}$ for $i \in [k]$. We refer to the graph consisting of these k disjoint parts as G^{Σ}. We shall review the properties of this graph after defining the remaining graphs related to **Multi-HPH**.

Before moving on, a quick remark is in order. RS graphs have been used extensively in the last decade for streaming lower bounds (see Section 4.2). However, *all* prior work on RS graphs that we are aware of has used them for encoding Boolean strings. For instance, [AR20] uses RS graphs to encode input sets to the *set disjointness* communication problem, and [CKP+21] uses them for encoding their direct-sum BHH problem (outlined earlier). To obtain more "complex" structures such as permutations, [CKP+21] further defined Boolean operations of RS graphs like $\vee$, $\wedge$, and $\oplus$ and used them in conjunction with sorting networks (we shall describe this connection shortly also). This is precisely the source of the $\Theta(\log n)$ loss in the parameters of permutation hiding graphs in [CKP+21] that we discussed earlier.

Despite its simplicity, our new construction—inspired by [A22], which still encoded a Boolean string in the RS graph but in a similar fashion—is directly encoding a permutation matrix inside the RS graph, which allows for encoding more complex structures, without having to pay *too much* on the density of the resulting graph.

Simple permutation graphs and (L, M, Γ)

We will create a new graph $H = H(G^{\Sigma}, L, M, \Gamma)$ out of G^{Σ}. For the purpose of this part, we only care about the *vertices* of G^{Σ} and not its edges (this will be crucial for the reduction). We shall not go into the rather tedious definition of the graph H here and instead simply mention its main properties (see also Figure 3 for an illustration):

- The graph H has asymptotically the same number of vertices as G^{Σ} and in particular includes an extra set of vertices $S = [r/2] \times [b]$ on its "left" most part layer. The extra edges inserted to H at this step form perm perfect matchings between the layers of H.

- If we start from the a-th group of vertices in S for $a \in [r/2]$, namely, $(a, *) \in S$, and follow the edges of the graph H, we shall be moving according to the hyperedge M_a across matchings $M^{\text{rs}}_{\ell_i} \otimes \Sigma^{(i)}$ for $i \in [k]$ in a way that we will end at the same block of vertices $(a, *)$ in the last layer of H; the mapping between the groups $(a, *) \in S$ and $(a, *)$ in the last layer will now be equal to $\gamma^\star_a \circ \gamma_a$.

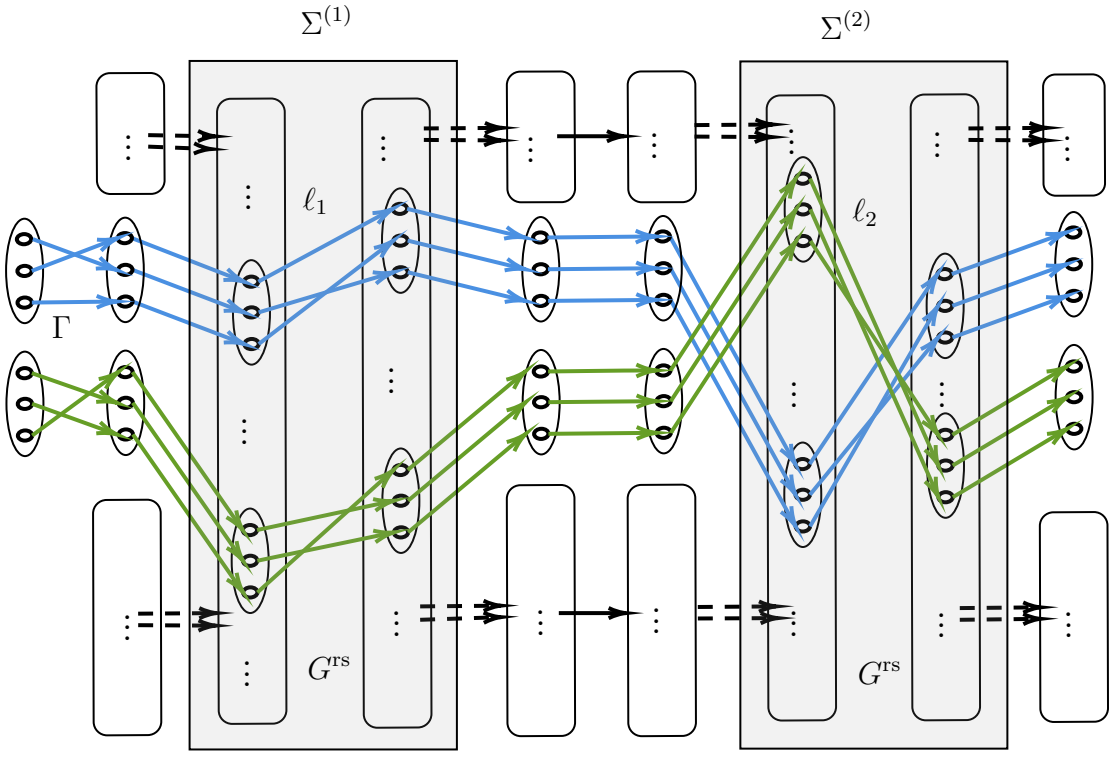


Figure 3: An illustration of the graph $H = H(G^\Sigma, L, M, \Gamma)$ for some unspecified parameters.

With the properties above, one can see that the graph H is in fact a permutation graph for the permutation induced[7] by the permutation vector $\Gamma^* \circ \Gamma \in (S_b)^{r/2}$ in **Multi-HPH**. Thus, learning the hidden permutation of the permutation graph H is equivalent to figuring out whether the answer to **Multi-HPH** was Γ_{Yes} or Γ_{No}.

We can now conclude the following. Given Σ, the players in **Multi-HPH** can create the graph G^Σ without any communication (as each player $Q^{(i)}$ will be responsible for creating $G_{\text{rs}} \otimes \Sigma^{(i)}$). The referee can also create the edges of the graph H given the inputs (L, M, Γ) as they *do not depend* on Σ. This in turn implies that a single-pass s-space streaming algorithm for learning the hidden permutation of the permutation graph H can be used as a $(k \cdot s)$-communication protocol for **Multi-HPH**. Given our lower bound in Theorem 2 for **Multi-HPH** (for appropriate choices of s), this implies that the resulting distribution we have on the graphs H is a permutation hiding generator with the important *caveat* that it can only hide permutation vectors, as opposed to arbitrary permutations.

[7] We can interpret each permutation vector in $(S_b)^{r/2}$ as a permutation over $[r/2 \cdot b]$, wherein for every $a \in [r/2]$, the elements $[(a-1) \cdot b + 1 : a \cdot b]$ are permuted according to the a-the permutation of the vector. We emphasize that while every permutation vector leads to a permutation, the converse is not true, and this is in fact going to play an important role in our arguments.

From Permutation Vectors to Arbitrary Permutations

At this point, we have already constructed a generator for permutation vectors. This generator can be seen as a middle ground between the *set hiding* generators of [CKP+21] (which are considerably weaker as they can hide limited permutations, roughly corresponding to the $b = 2$ case) and permutation hiding generators (which are considerably stronger, as they can hide every permutation).

Our last step to permutation hiding generators is inspired by a nice strategy of [CKP+21] for going from their set hiding graphs to their permutation hiding graphs. They show that one can use *sorting networks* (see Section 4.3), and in particular, the celebrated *AKS sorters* [AKS83] to compute any arbitrary permutation as concatenation of $\Theta(\log n)$ "matching permutations", namely, permutations that can only change two previously fixed pairs of elements with each other. This allows them to obtain a permutation hiding generator from $\Theta(\log n)$ set hiding ones – this is precisely the source of the factor $\Theta(\log n)$ loss in the construction of [CKP+21].

For our purpose, we also use sorting networks but this time with larger comparators (typically called *b-sorters*), wherein each comparator can sort b wires simultaneously (see Section 4.3). We can then use an extension of the sorting network of [AKS83] to b-sorter networks due to [Chv92] with depth $O(\log_b n)$ instead (see also [PP89] and Appendix D). Finally, we show that our generator for hiding permutation vectors can "simulate" each layer of this network efficiently and use this to obtain a generator for every arbitrary permutation, by applying our generator for permutation vectors $O(\log_b n)$ times. By taking b to be a sufficiently small polynomial in n, we can ensure that $O(\log_b n) = O(1)$ and thus only pay a constant factor overhead when constructing our final permutation hiding generator for single-pass algorithms. Finally, b is still sufficiently small such that even though in our encoded RS graph, we essentially make the input graph sparser by a factor of b, the graph is still sufficiently dense to allow for our desired lower bound. The language used in Sections 6 and 7 is slightly different for readability. We work with permutations induced by permutation vectors, which we call simple permutations (see Definition 4.2) directly.

3.2.2 Permutation Hiding for Multi-Pass Algorithms

The last step of our approach is to go from single-pass algorithms to multi-pass algorithms using the strong guarantee of permutation hiding generators and the power of back-and-forth communication between the players, but not the referee, in the **Multi-HPH** problem. While at a technical level this step still requires addressing several new challenges, at a conceptual level, it more or less mimics that of [CKP+21] without any particularly novel ideas.

The goal is now to construct a permutation hiding generator for p-pass algorithms, simply denoted by $\mathcal{G}_p(\cdot)$, from a generator for $(p-1)$-pass algorithms denoted similarly by $\mathcal{G}_{p-1}(\cdot)$. The main idea is still based on a reduction from **Multi-HPH** for inputs (Σ, L, M, Γ). A key point in our construction of G^{Σ} and H for single-pass algorithms is that the edges added by the referee, but certainly not the players, to H form different *matchings* $M_1, M_2, \ldots$ between *disjoint* sets of vertices of H. In particular, if we see H as a layered graph, some of these layers are constructed based on G^{Σ} and the edges the between remaining layers are perfect matchings.

To create our p-pass generator, we start with creating a graph H from (Σ, L, M, Γ) as described in the previous part. Then, for every one of the perfect-matching layers M_j described above, we replace those layers with a permutation hiding graph $\mathcal{G}_{p-1}(M_j)$; here, we consider the perfect matching M_j as a permutation over the vertices of the layer. This will increase the number of vertices, and the number of layers, in the entire graph by a constant factor. See Figure 4 for an illustration.

We can now claim that this new family of graphs is in fact a p-pass generator. At an intuitive

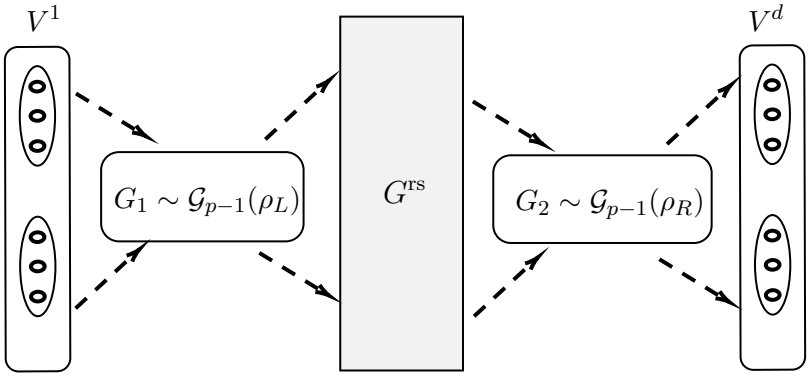


Figure 4: An illustration a (subgraph of a) p-pass generator from $(p-1)$-pass generator.

level, we can say that the first $(p-1)$ passes of the algorithm cannot figure (L, M, Γ) as it cannot determine the identity of each of the matchings $M_1, M_2, \ldots$, added to H by the referee. This is by the guarantee of the generator $\mathcal{G}_{p-1}$ against $(p-1)$ pass algorithms. Because of this, the problem "effectively" become the same as the single-pass case again, and we can perform a reduction from **Multi-HPH** as before. Making this intuition precise requires quite a lot of technical work, but as we stated earlier, it follows a similar pattern as that of [CKP+21] (plus certain simplifications using a similar argument in [AR20]), and we postpone their details to Section 7.

3.3 Proof Outline of Theorem 1: A Multi-Pass Lower Bound for Matchings

The proof is based on a standard reduction, e.g., in [AR20], from finding vertex-disjoint paths to bipartite matching (a similar reduction also appeared in [CKP+21]). Given the simplicity of this proof, we more or less provide its full details here for completeness.

Consider any permutation graph $G = (V, E)$ for some $\sigma \in S_m$. A simple observation is that for any pairs of vertices $s_1, s_2 \in \textsc{First}_{[m]}(G)$, their corresponding paths to $\sigma(s_1), \sigma(s_2) \in \textsc{Last}_{[m]}(G)$ should be vertex disjoint. Suppose not, and let $v \in V$ be one intersecting vertex; then, $s_1 \rightsquigarrow v \rightsquigarrow \sigma(s_1)$ and $s_1 \rightsquigarrow v \rightsquigarrow \sigma(s_2)$ should be some paths in G, violating the permutation graph property of G. We use this observation crucially in our proof.

Fix a permutation hiding generator $\mathcal{G} := \mathcal{G}_{m,n,s,p,\delta}$ for any integers $m, p \geqslant 1$ and parameters n, s, δ as in Theorem 3. For any graph G in the support of any of the distributions output by $\mathcal{G}$, let $S = \textsc{First}_{[m/2]}(G)$ and $T = \textsc{Last}_{[m/2]}(G)$. Now,

- By taking $\sigma_=$ to be the identity permutation, we can ensure that in any $G \sim \mathcal{G}(\sigma_=)$, there are $m/2$ vertex-disjoint paths from S to T in G;
- But, by taking $\sigma_\times$ to be the "cross identity", namely, a one mapping $[m/2]$ to $[m/2+1 : m]$ identically and vice versa, we can ensure that in $G \sim \mathcal{G}(\sigma_\times)$, there are no vertex-disjoint paths from S to T in G.

The next step is to turn this vertex-disjoint path problem into a bipartite matching instance. Roughly speaking, this is done by turning these paths into augmenting paths for a canonical matching in the bipartite matching instance.

Consider the following function $\textsc{Bipartite}(G)$ that given a permutation graph G in the support of any distribution in $\mathcal{G}$ creates the following bipartite graph:

(*i*) Copy the vertices of G twice to obtain two sets V^{L} and V^{R}. Additionally, add two new sets S_0 and T_0 to the graph.

(*ii*) For every vertex $v \in G$, connect $v^{\text{L}} \in V^{\text{L}}$ to $v^{\text{R}} \in V^{\text{R}}$ (refer to these edges as a matching M).

(*iii*) For every edge $(u, v) \in G$, connect $u^{\text{L}} \in V^{\text{L}}$ to $v^{\text{R}} \in V^{\text{R}}$. Additionally, for every vertex $s_i \in S$ of G, connect $s_i \in S_0$ to $s_i^{\text{R}} \in V^{\text{R}}$ and for every vertex $t_i \in T$ of G, connect $t_i \in T_0$ to $t_i^{\text{L}} \in V^{\text{L}}$.

It is not hard to see that in this graph, any augmenting path for the matching M has to start from S_0 and end in T_0. In addition, the structure of the graph, plus the alternating nature of an augmenting path, forces the path to basically follow the copies of edges in G from S to T. This will turn imply that:

- When $G \sim \mathcal{G}(\sigma_{=})$, the matching M in $\textsc{Bipartite}(G)$ has $m/2$ vertex-disjoint augmenting paths, and thus can be augmented to a perfect matching of size $n + m/2$;
- On the other hand, when $G \sim \mathcal{G}(\sigma_{\times})$, the matching M in $\textsc{Bipartite}(G)$ has no augmenting paths and is thus a maximum matching of size n.

Given that streaming algorithms running on G can generate $\textsc{Bipartite}(G)$ "on the fly", this is sufficient to prove that any streaming algorithm that can determine whether the input graph has a perfect matching of size $n + m/2$ or its largest matching is of size n, would also distinguish between the distributions $\mathcal{G}(\sigma_{=})$ and $\mathcal{G}(\sigma_{\times})$. But, by the indistinguishability of these distributions in Theorem 3, we obtain that no p-pass s-space algorithm can solve the underlying matching problem. We can now instantiate the parameters of Theorem 3 as follows.

- The parameter ε of Theorem 1 is obtained via:

$$(1 - \varepsilon) \cdot \left(n + \frac{m}{2}\right) = n,$$

  which implies that $\varepsilon = \Theta(m/n)$.

- The number of vertices of $\textsc{Bipartite}(G)$ is $n + m/2$ on each side and thus $\Theta(n)$ in general.

- The number of passes of the algorithm needs to be at least

$$p = \Omega\Big(\frac{\log\left((n + m/2)/m\right)}{\log(1/\alpha \cdot \beta^2)}\Big) = \Omega\Big(\frac{\log\left(1/\varepsilon\right)}{\log\left(1/\alpha\beta\right)}\Big).$$

- The bound on the space s of the algorithms needs to be

$$s = o(m^{1+\beta/2}) = o((\alpha \cdot \beta^2)^{p\cdot(1+\beta/2)} \cdot (n + m/2)^{1+\beta/2}) = o(\varepsilon^2 \cdot (n + m/2)^{1+\beta/2}).$$

This then concludes the proof of Theorem 1.

4 Preliminaries

4.1 Notation

Graphs

For any graph $G = (V, E)$, we use $n := |V|$ to denote the number of vertices. For an edge $e = (u, v)$, we use $V(e)$ to denote the vertices incident on e. When G is a directed graph, for any vertices

$s, t \in V$, we write $s \to t$ to mean there is an edge from s to t and $s \rightsquigarrow t$ to mean there is a path from s to t in G.

We sometimes say that two (or more) disjoint sets S and T of a graph G with size $m = |S| = |T|$ are identified by the set $[m]$ to mean that any integer $i \in [m]$ can be used to refer to both the i-th vertex of S as well as T, when it is clear from the context (e.g., we say that for any $i \in [m]$, connect vertex $i \in S$ to vertex $i \in T$).

Tuples and matrices

For any m-tuple $X := (x_1, \ldots, x_m)$, and any integer $i \in [m]$, we define

$$X_{<i} := (x_1, \ldots, x_{i-1}) \qquad \text{and} \qquad X_{-i} := (x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_m).$$

We can further define $X_{>i}$ and $X_{\leqslant i}$ analogously and extend these definitions to vectors as well.

For a $A \in \mathbb{R}^{m \times n}$, we use $A_{i,*}$ to denote the i-th row of the matrix A and $A_{*,j}$ to denote the j-th column. We denote the sub-matrix of A on rows in $S \subseteq [m]$ and $T \subseteq [n]$ by $A_{S,T}$, and similarly use $A_{S,*}$ and $A_{*,T}$ to only limited the rows or and columns to S and T, respectively.

Permutations

We use S_m to denote the symmetric group of permutations over the set $[m]$. We use $\sigma_{\text{ID}}(m) \in S_m$ to denote the identity permutation on $[m]$. For any permutation matrix $\Sigma \in (S_b)^{t \times r}$, we use $\sigma_{i,j} \in S_b$ to denote the permutation at row i and column j for $i \in [t], j \in [r]$.

4.2 Bipartite Ruzsa-Szemerédi-Graphs

Let $G = (V, E)$ be an undirected graph, and $M \subseteq E$ be a matching in G. We say that M is an **induced matching** iff the subgraph of G induced on the vertices of M is the matching M itself; in other words, there are no other edges between the vertices of this matching.

Definition 4.1 (Ruzsa-Szemerédi Graphs [RS78])**.** For any integers $r, t \geqslant 1$, a *bipartite* graph $G_{\text{rs}} = (L_{\text{rs}} \cup R_{\text{rs}}, E_{\text{rs}})$ is called a bipartite (r, t)-**Ruzsa-Szemerédi graph** (RS graph for short) iff its edge-set E can be partitioned into t *induced* matchings $M_1^{\text{rs}}, \ldots, M_t^{\text{rs}}$, each of size r.

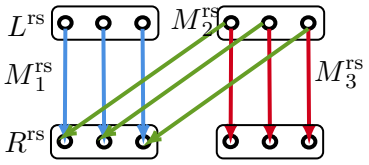


Figure 5: An illustration of simple RS-Graph construction.

It is easy to construct bipartite (r, t)-RS graphs with n vertices on each side and parameter $t = (n/r)^2$ for any $r \geqslant 1$ (see Figure 5). More interestingly, and quite surprisingly, one can create much denser RS graphs as well; for instance, the original construction of these graphs due to Ruzsa and Szemerédi [RS78] create graphs with $r = n/2^{\Theta(\sqrt{\log n})}$ and $t = \Omega(n)$. Yet another construction is that of [FLN+02, GKK12] that shows that already for $r < n/2$, we can have $t = n^{\Omega(1/\log\log n)}$. See [BLM93, FLN+02, Alo02, TV06, AS06, AMS12, GKK12, FHS17, AB19, KKTY21] for various other constructions and applications of RS graphs. At the same time, proving upper bounds on the density of RS graphs turned out to be a notoriously difficult question (see, e.g. [Gow01, FHS17, CF13]) and the best known upper bounds, even for $r = \Omega(n)$, only imply that $t < n/2^{O(\log^* n)}$ [Fox11, FHS17].

A line of work initiated by Goel, Kapralov, and Khanna [GKK12] have used different constructions of RS graphs to prove lower bounds for graph streaming algorithms [GKK12, Kap13, Kon15, AKLY16, AKL17, CDK19, AR20, Kap21, AB21, CKP+21, KN21, A22] (very recently, they have also been used in [ABKL23] for providing graph streaming *algorithms* for the matching problem).

Notation for RS-Graphs. The number of vertices in one partition of the bipartite graph $G_{\text{rs}} = (L_{\text{rs}} \cup R_{\text{rs}}, E_{\text{rs}})$ is denoted by n^{rs} (the total number of vertices is $2n^{\text{rs}}$). The two sets L_{rs} and R_{rs} are both identified by the set $[n^{\text{rs}}]$. For each matching M_i^{rs} for $i \in [t]$, the edges in the matching are identified by the set $[r]$. We also use e to iterate over the edges in E_{rs}, or point to any arbitrary edge in E_{rs}. Given any $e \in E_{\text{rs}}$, we use $\textsc{left}(e) \in [n^{\text{rs}}]$ and $\textsc{right}(e) \in [n^{\text{rs}}]$ to denote the vertices in L_{rs} and R_{rs} that edge e is incident on, respectively.

For any (r,t)-Ruzsa-Szemerédi graph G_{rs}, we use $\alpha, \beta \in [0,1]$ to denote the following parameters.

$$\alpha := \frac{r}{n^{\text{rs}}} \quad \beta := \frac{\log(t)}{\log(n^{\text{rs}})}. \tag{6}$$

That is, G_{rs} has $(n^{\text{rs}})^{\beta}$ induced matchings of size $\alpha \cdot n^{\text{rs}}$ each.

4.3 Sorting Networks with Large Comparators

Following [CKP+21], we use sorting networks (see, e.g. [Knu97, Section 5.3.4]) in our constructions. The key difference is that we need sorting networks that work with larger comparators, typically called *b-sorters* (i.e., each sorter can sort $b > 2$ wires in one step as a primitive). We refer the reader to [PP89, Chv92] for more information on sorting networks with b-sorters (see Figure 6).

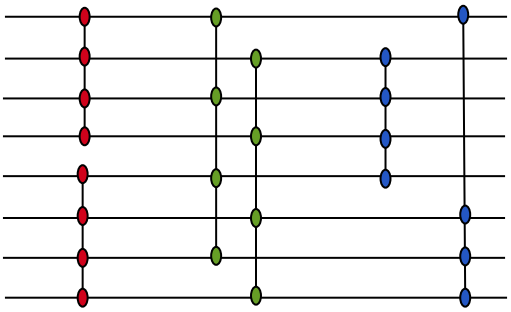


Figure 6: A sorting network with $b = 4$ size sorters, $m = 8$ wires, and depth $d = 3$.

We shall use the following result from [Chv92] that is a generalization of the celebrated *AKS sorter* of [AKS83] with $O(\log n)$ depth to networks with b-sorters with $O(\log_b n)$ depth instead[8]. We present this result directly in the language we shall use later in this paper, but the equivalence in terms of sorting networks is straightforward.

Definition 4.2. Given an equipartition $\mathcal{P}$ of the set $[m]$ into r groups $P_1, P_2, \ldots, P_r$ of size $b = m/r$ each, a permutation $\sigma \in S_m$ is said to be **simple** on partition $\mathcal{P}$ if for each group P_j with $j \in [r]$, for any element $i \in [m]$ belonging to group P_j, we have that $\sigma(i) \in P_j$ also.

Informally, a simple permutation permutes elements only inside the groups in the partition $\mathcal{P}$.

[8] Although we note that given the range of parameters used in our paper, namely, $b = n^{\Omega(1)}$, we can alternatively use the much simpler $O(\log_b^2(n))$-depth construction of [PP89] as well to the same effect; see Appendix D.

Proposition 4.3 ([Chv92]; see also [AKS83]). *There exists an absolute constant $c_{\text{SORT}} > 0$ such that the following is true. For every pair of integers $r, b \geqslant 1$ and $m = r \cdot b$, there exists*

$$d_{\text{SORT}} = d_{\text{SORT}}(r, b) = c_{\text{SORT}} \cdot \log_b(m)$$

fixed equipartitions of $[m]$ into $\mathcal{P}_1, \ldots, \mathcal{P}_{d_{\text{SORT}}}$, each one consisting of r sets of size b, with the following property. Given any permutation $\sigma \in S_m$, there are d_{SORT} permutations $\gamma_1, \ldots, \gamma_{d_{\text{SORT}}}$ where for every $i \in [d_{\text{SORT}}]$, γ_i is simple on partition $\mathcal{P}_i$ so that we have $\sigma = \gamma_1 \circ \cdots \circ \gamma_{d_{\text{SORT}}}$.

To interpret this result in terms of sorting networks on m wires, think of each $\mathcal{P}_i$ for $i \in [d_{\text{SORT}}]$ as one layer of the sorting network and each b-sorter in this layer as one component of $\mathcal{P}_i$ (see Figure 6). We have provided a proof of a weaker version of Proposition 4.3 when $b = 2$ with a larger depth $d_{\text{SORT}} = O(\log^2(m))$ for intuition in Appendix D. The proof of Proposition 4.3 with $d_{\text{SORT}} = c_{\text{SORT}} \log_b(m)$ and $b > 2$ is quite involved, and the details can be found in [Chv92].

Simple Permutations and Permutation Vectors

In Section 5, we work with permutation vectors from $(S_b)^{r/2}$ with $r = 2m/b$ and in Sections 6 and 7, we predominantly work with simple permutations. We will establish the bijection between the two for any fixed partition here. Let equipartition Lex of $[m]$ be the partition that splits the elements lexicographically into groups of size b. That is, for each $i \in [m/b]$:

$$\mathsf{Lex}_i = \{j \mid (i-1) \cdot b + 1 \leqslant j \leqslant i \cdot b\}.$$

Let $S_m^{\text{SIM}} \subset S_m$ be the set of all permutations which are simple on partition Lex.

Observation 4.4. *There is a bijection* $\text{VEC} : S_m^{\text{SIM}} \to (S_b)^{m/b}$.

Proof. We know that partition Lex lexicographically groups the elements of $[m]$ into groups of size b. For any $\rho \in S_m^{\text{SIM}}$, we define $\text{VEC}(\rho)$ as follows. For $i \in [m/b]$, let $\gamma_i \in S_b$ be,

$$\gamma_i(j) = \rho((i-1)b + j) - (i-1)b$$

for each $j \in [b]$. As $\rho((i-1)b + j)$ belongs to the same partition as $(i-1)b + j$ for $j \in [b]$, γ_i is a permutation in S_b. Then, $\text{VEC}(\rho) = (\gamma_1, \gamma_2, \ldots, \gamma_{m/b})$. ∎

Note that Observation 4.4 is applicable for any set of simple permutations for a fixed partition, but we will work mainly with Lex.

Notation. For any permutation $\rho \in S_m^{\text{SIM}}$, we use $\text{VEC}(\rho)$ to denote the corresponding element from $(S_b)^{m/b}$. For any permutation vector $\Gamma = (\gamma_1, \gamma_2, \ldots, \gamma_{m/b}) \in (S_b)^{m/b}$, we use $\text{JOIN}(\Gamma)$ to denote the corresponding element of S_m^{SIM}, as the tuple is combined into a single larger permutation. (This is just the inverse of bijection VEC from Observation 4.4.)

4.4 Streaming Algorithms

For the purpose of our lower bounds, we shall work with a more powerful model than what is typically considered when designing streaming algorithms (this is the common approach when proving streaming lower bounds; see, e.g. [GM08, LNW14, BGW20]). In particular, we shall define streaming algorithms similar to branching programs as follows and then point out the subtle differences with what one typically expect of a streaming algorithm.

Definition 4.5 (**Streaming algorithms**)**.** For any integers $m, p, s \geqslant 1$, we define a p-pass s-space streaming algorithm A working on a length-m stream $\sigma = (\sigma_1, \ldots, \sigma_m)$ with entries from a universe U as follows:

(*i*) The algorithm A has access to a read-only tape of uniform random bits r from a finite, but arbitrarily large range R without having to pay for the cost of storing these bits.

(*ii*) There is a function $f_A : \{0,1\}^s \times U \times R \to \{0,1\}^s$ that updates the state of the algorithm as follows. The algorithm starts with the state $S := 0^s$, and for $i \in [m]$, whenever A reads σ_i in the stream during its p passes, it updates its current state S to $S \leftarrow f_A(S, \sigma_i, r)$ (the algorithm is computationally unbounded when computing its next state).

(*iii*) At the end of the last pass, the algorithm outputs the answer as a function of its state S and random bits r.

(We note that this model is non-uniform and is defined for each choice of m, p, s individually.)

Let us point out two main differences with what one may expect of streaming algorithms. Firstly, we allow our streaming algorithms to do an unbounded amount of work using an unbounded amount of space *between* the arrival of each stream element; we only bound the space in transition between two elements. Secondly, we do not charge the streaming algorithms for storing random bits.

Clearly, any lower bound proven for streaming algorithms in Definition 4.5 will hold also for more restrictive (and "algorithmic") definitions of streaming algorithms. We shall note that however almost all streaming lower bounds we are aware of directly work with this definition and thus we claim no strengthening in proving our lower bounds under this definition; rather, we merely use this formalism to carry out the reductions in our arguments formally.

For any p-pass algorithm A, $q \in [p]$, and input distribution μ, let $\textsc{mem}_A^q(\mu)$ denote the memory state of A after q passes plus the content of its random tape on an input sampled from μ. Note that $\textsc{mem}_A^q(\mu)$ is a random variable depending on randomness of A as well as μ.

Definition 4.6 (δ-Indistinguishable Distributions)**.** For any $\delta \in [0,1]$, the two distributions μ, ν are said to be δ-indistinguishable for p-pass s-space streaming algorithms if for every such algorithm A,

$$\|\textsc{mem}_A^p(\mu) - \textsc{mem}_A^p(\nu)\|_{\text{tvd}} \leqslant \delta.$$

Finally, we have the following standard hybrid argument for multi-pass streaming algorithms (see, e.g. [CKP+21, AN21]). We present its short proof for completeness.

Proposition 4.7 (c.f. [CKP+21])**.** *Let ℓ be a positive integer and $\delta \in \mathbb{R}^\ell_{\geqslant 0}$ denote ℓ parameters. Let $(\mu_1, \nu_1), (\mu_2, \nu_2), \ldots, (\mu_\ell, \nu_\ell)$ be ℓ pairs of distributions, such that for every $i \in [\ell]$, μ_i and ν_i are δ_i-indistinguishable for p-pass s-space streaming algorithms. Then, $\mu := (\mu_1, \ldots, \mu_\ell)$ and $\nu := (\nu_1, \ldots, \nu_\ell)$ are $\|\delta\|_1$-indistinguishable for p-pass s-space streaming.*

Proof. For any $i \in [\ell]$, define the hybrid distribution:

$$h_i := (\nu_1, \ldots, \nu_i, \mu_{i+1}, \ldots, \mu_\ell).$$

This way, $h_0 = \mu$ and $h_\ell = \nu$. Consider any p-pass s-space streaming algorithm A. We prove that for every $i \in [\ell]$,

$$\|\text{MEM}_A^p(h_{i-1}) - \text{MEM}_A^p(h_i)\|_{\text{tvd}} \leqslant \delta_i,$$

by turning A into a p-pass s-space streaming algorithm B for distinguishing between μ_i and ν_i. Algorithm B, given a stream σ from either μ_i or ν_i, is defined as follows:

(*i*) Sample the inputs $\sigma_1, \ldots, \sigma_{i-1} \sim \nu_1, \ldots, \nu_{i-1}$ and $\sigma_{i+1}, \ldots, \sigma_\ell \sim \mu_{i+1}, \ldots, \mu_\ell$. This sampling is free of charge for the algorithm B by Definition 4.5.

(*ii*) Run A on the input $(\sigma_1, \ldots, \sigma_{i-1}, \sigma, \sigma_{i+1}, \ldots, \sigma_\ell)$ in p passes and s space.

We thus have,

$$\|\text{MEM}_A^p(h_{i-1}) - \text{MEM}_A^p(h_i)\|_{\text{tvd}} = \|\text{MEM}_B^p(\mu_i) - \text{MEM}_B^p(\nu_i)\|_{\text{tvd}} \leqslant \delta_i,$$

by the ε_i-indistinguishability of μ_i and ν_i for p-pass s-space streaming algorithm B. The final result now follows from triangle inequality. ∎

5 The Multi Hidden Permutation Hypermatching Problem

We prove Theorem 2 on the communication cost of **Multi-HPH** in this section. For the convenience of the reader, we restate the definition of this communication game and our theorem below.

Problem (Restatement of Problem 1)**.** For any integers $r, t, b, k \geqslant 1$, $\textbf{Multi-HPH}_{r,t,b,k}$ is a distributional $(k+1)$-communication game, consisting of k players plus a referee, defined as:

(*i*) Let $\Gamma_{\text{Yes}}, \Gamma_{\text{No}} \in (S_b)^{r/2}$ be two arbitrary tuples of permutations known to all $(k+1)$ parties. We refer to Γ_{Yes} and Γ_{No} as the **target tuples**.

(*ii*) We have k players $Q^{(1)}, \ldots, Q^{(k)}$ and for all $i \in [k]$ the i-th player is given a permutation matrix $\Sigma^{(i)} \in (S_b)^{t \times r}$ chosen independently and uniformly at random.

(*iii*) The referee receives k indices $L := (\ell_1, \ldots, \ell_k)$ each picked uniformly and independently from $[t]$ and a hypermatching $M \subseteq [r]^k$ with $r/2$ hyperedges picked uniformly.

Additionally, let the permutation vector $\Gamma^* = \left(\gamma_1^\star, \gamma_2^\star, \ldots, \gamma_{r/2}^\star\right) \in (S_b)^{r/2}$ be defined as,

$$\forall a \in [r/2] \;\; \gamma_a^\star := \sigma^{(1)}_{\ell_1, M_{a,1}} \circ \cdots \circ \sigma^{(k)}_{\ell_k, M_{a,k}},$$

where $M_{a,i}$ for $a \in [r/2]$ and $i \in [k]$ refers to the i-th vertex of the a-th hyperedge in M.

The referee is also given another permutation vector $\Gamma = (\gamma_1, \gamma_2, \ldots, \gamma_{r/2})$ sampled from $(S_b)^{r/2}$ conditioned on $\Gamma^* \circ \Gamma$ being equal to either Γ_{Yes} or Γ_{No}.

The players $Q^{(1)}, \ldots, Q^{(k)}$ can communicate with each other by writing on a shared board visible to all parties with possible back and forth and in no fixed order (the players' messages are functions of their inputs and the board). At the end of the players' communication, the referee can use all these messages plus its input and outputs whether $\Gamma^* \circ \Gamma$ is Γ_{Yes} or Γ_{No}.

Theorem (Restatement of Theorem 2)**.** *For any $t \geqslant 1$, $b \geqslant 2$, and sufficiently large $r, k \geqslant 1$, any communication protocol for* **Multi-HPH**$_{r,t,b,k}$*, for any choice of target tuples, with at most s bits of total communication for s satisfying*

$$k \cdot \log(r \cdot t) \leqslant s \leqslant 10^{-3} \cdot (r \cdot t)$$

can only succeed with probability at most

$$\frac{1}{2} + r \cdot O\Big(\frac{s}{r \cdot t}\Big)^{k/32}.$$

When going through the proof, we shall condition on several probabilistic events and properties at different points, and from thereon continue all our analysis conditioned on them. We will mark each of these steps clearly in the corresponding subsection as a "conditioning step" (typically as the conclusion of the subsection), in order to help the reader keep track of them.

5.1 Part One: Setup and the Basic Problem

We begin our proof of Theorem 2 in this subsection by defining the "basic" problem we will need to focus on to prove this result. This will set up the stage for the main parts of the proof in the subsequent subsections. We start with some notation.

Notation. We fix $t \geqslant 1$, $b \geqslant 2$, and sufficiently large choice of integers $r, k \geqslant 1$ and consider the **Multi-HPH**$_{r,t,b,k}$ problem (henceforth, denoted simply by **Multi-HPH**) with any two arbitrarily target tuples Γ_{Yes} and Γ_{No}. We further define

$$\Sigma := (\Sigma^{(1)}, \ldots, \Sigma^{(k)}),$$

to denote the input to all players $Q^{(1)}, \ldots, Q^{(k)}$.

From now on, fix a **deterministic** protocol π for **Multi-HPH** and let $\Pi = (\Pi_1, \ldots, \Pi_k)$ denote the transcript of the protocol, where Π_i is the messages communicated the player $Q^{(i)}$ for $i \in [k]$. As per the theorem statement, we assume that π communicates s bits in total.

Finally, throughout this section, we use sans-serif letters to denote the random variables for corresponding objects, e.g., $\mathsf{\Sigma}$ and $\mathsf{\Pi}$ for the input Σ and messages Π of players, and $\mathsf{L}, \mathsf{M}, \mathsf{\Gamma}$ for the input of referee (L, M, Γ). Moreover, we may use random variables and their distributions interchangeably when the meaning is clear from the context.

5.1.1 Removing the Role of $\Gamma_{\text{Yes}}, \Gamma_{\text{No}}$, and Γ

We are now ready to start the proof. We can interpret the goal of the referee in **Multi-HPH** as follows: Given (Π, L, M), the final input of the referee is the tuple of permutations Γ chosen uniformly at random from one of the following two distributions:

$$(\mathsf{\Gamma}^{*-1} \circ \Gamma_{\text{Yes}} \mid \Pi, L, M) \qquad \text{or} \qquad (\mathsf{\Gamma}^{*-1} \circ \Gamma_{\text{No}} \mid \Pi, L, M);$$

here, and throughout, Γ^{*-1} is interpreted as taking the inverse of each permutation in the tuple.

Thus, given one sample Γ from a uniform mixture of the above two distributions, the referee needs to determine which distribution Γ was sampled from. This, together with Fact A.8 (on success probability of distinguishing distributions with one sample), implies that the probability of success of the referee is equal to:

$$\Pr_{\Pi, L, M, \Gamma}(\pi \text{ succeeds}) = \frac{1}{2} + \frac{1}{2} \cdot \mathop{\mathbb{E}}_{\Pi, L, M} \|(\mathsf{\Gamma}^{*-1} \circ \Gamma_{\text{Yes}} \mid \Pi, L, M) - (\mathsf{\Gamma}^{*-1} \circ \Gamma_{\text{No}} \mid \Pi, L, M)\|_{\text{tvd}}. \tag{7}$$

Our plan is to bound the RHS of Eq (7). To do so, we are going to do this indirectly by proving that both distributions considered in this equation are quite close to the uniform distribution. Another simple step also allows us to entirely ignore the choice of Γ_{Yes} and Γ_{No} and simply consider the distribution of Γ^* itself. These parts are captured in the following claim.

Claim 5.1. *For any choice of* Π, L, M*,*

$$\|(\Gamma^{*-1} \circ \Gamma_{\text{Yes}} \mid \Pi, L, M) - (\Gamma^{*-1} \circ \Gamma_{\text{No}} \mid \Pi, L, M)\|_{\text{tvd}} \leqslant 2 \cdot \|(\Gamma^* \mid \Pi, L, M) - \mathcal{U}_{S_b^{r/2}}\|_{\text{tvd}},$$

where $\mathcal{U}_{S_b^{r/2}}$ *is the uniform distribution over tuples in* $S_b^{r/2}$*.*

Proof. By the triangle inequality,

$$\text{LHS of the claim} \leqslant \|(\Gamma^{*-1} \circ \Gamma_{\text{Yes}} \mid \Pi, L, M) - \mathcal{U}_{S_b^{r/2}}\|_{\text{tvd}} + \|\mathcal{U}_{S_b^{r/2}} - (\Gamma^{*-1} \circ \Gamma_{\text{No}} \mid \Pi, L, M)\|_{\text{tvd}}.$$

For the first term above, we have,

$$\begin{aligned}
\|(\Gamma^{*-1} \circ \Gamma_{\text{Yes}} \mid \Pi, L, M) - \mathcal{U}_{S_b^{r/2}}\|_{\text{tvd}} &= \frac{1}{2} \cdot \sum_{\Gamma \in S_b^{r/2}} \left| \Pr\left(\Gamma^{*-1} \circ \Gamma_{\text{Yes}} = \Gamma \mid \Pi, L, M\right) - \left(\frac{1}{b!}\right)^{r/2} \right| \\
&\qquad \text{(by the definition of TVD in Eq (30))} \\
&= \frac{1}{2} \cdot \sum_{\Gamma \in S_b^{r/2}} \left| \Pr\left(\Gamma^* = (\Gamma \circ \Gamma_{\text{Yes}}^{-1})^{-1} \mid \Pi, L, M\right) - \left(\frac{1}{b!}\right)^{r/2} \right| \\
&\qquad \text{(each tuple of permutations has a unique inverse by taking inverse of each of its permutations)} \\
&= \frac{1}{2} \cdot \sum_{\Gamma^* \in S_b^{r/2}} \left| \Pr\left(\Gamma^* = \Gamma^* \mid \Pi, L, M\right) - \left(\frac{1}{b!}\right)^{r/2} \right| \\
&\qquad \text{(as there is a one-to-one mapping between } S_b^{r/2} \text{ and } (\Gamma \circ \Gamma_{\text{Yes}}^{-1})^{-1} \text{ when } \Gamma \text{ ranges over } S_b^{r/2}) \\
&= \|(\Gamma^* \mid \Pi, L, M) - \mathcal{U}_{S_b^{r/2}}\|_{\text{tvd}}. \\
&\qquad \text{(again, by the definition of TVD in Eq (30))}
\end{aligned}$$

Applying the same calculation (by replacing Γ_{Yes} with Γ_{No}) to the second term above concludes the proof of this claim. ∎

Plugging in the bounds we obtained in Claim 5.1 in Eq (7), we obtain that

$$\Pr_{\Pi, L, M, \Gamma}(\pi \text{ succeeds}) \leqslant \frac{1}{2} + \mathop{\mathbb{E}}_{\Pi, L, M} \|(\Gamma^* \mid \Pi, L, M) - \mathcal{U}_{S_b^{r/2}}\|_{\text{tvd}}. \tag{8}$$

The rest of the proof is dedicated to bounding the RHS of Eq (8). Notice this at this point, we have entirely removed the role of $\Gamma_{\text{Yes}}, \Gamma_{\text{No}}$, and Γ. Thus, from now on, we can solely focus on the power of players in **Multi-HPH** in changing the distribution of Γ^* from its original distribution which is uniform over $(S_b)^{r/2}$.

A remark about the importance of Γ is in order. All the inputs in Problem 1, barring Γ are chosen uniformly at random and independently from their support sets. This is crucial to our proof, as we will see in the later parts of this section. Without Γ, (say if we set $\Gamma^* = \Gamma_{\text{Yes}}$ or Γ_{No}) the original distribution of Γ^* would not be uniform, and we cannot bound the RHS of Eq (8).

5.1.2 From the Hypermatching to a Single Hyperedge

We now further simplify our task in proving an upper bound on the RHS of Eq (8). Recall that $\Gamma^* = (\gamma_1^\star, \ldots, \gamma_{r/2}^\star)$. By the (weak) chain-rule property of total variation distance in Fact A.9,

$$\mathop{\mathbb{E}}_{\Pi,L,M} \|(\Gamma^* \mid \Pi, L, M) - \mathcal{U}_{S_b^{r/2}}\|_{\text{tvd}} \leqslant \sum_{a=1}^{r/2} \mathop{\mathbb{E}}_{\Pi,L,M} \mathop{\mathbb{E}}_{\Gamma^*_{<a}|\Pi,L,M} \|(\gamma_a^\star \mid \Pi, L, M, \Gamma^*_{<a}) - \mathcal{U}_{S_b}\|_{\text{tvd}}. \tag{9}$$

Our goal is thus to bound each individual term in the RHS of Eq (9). To continue we need the following definitions. For any choice of Π, L, M, and integers $a \in [r/2]$, $i \in [k]$, and $j \in [t]$, define:

$$\begin{aligned}
M_a &:= (M_{a,1}, \ldots, M_{a,k}) && \text{(the } a\text{-th hyperedge in } M\text{)}\\
M_{<a} &:= (M_1, \ldots, M_{a-1}) && \text{(the first } a-1 \text{ hyperedges in } M\text{)}\\
\Sigma^{(i)}_{j,M_{<a}} &:= (\sigma^{(i)}_{j,M_{1,i}}, \ldots, \sigma^{(i)}_{j,M_{a-1,i}}) && \text{(the } a-1 \text{ entries of } \Sigma^{(i)}_{j,*} \text{ indexed by } M_{<a}\text{)}\\
\Sigma^{(i)}_{*,M_{<a}} &:= (\Sigma^{(i)}_{1,M_{<a}}, \ldots, \Sigma^{(i)}_{t,M_{<a}}) && \text{(the input of } Q^{(i)} \text{ on hyperedges in } M_{<a}\text{)}\\
\Sigma_{*,M_{<a}} &:= (\Sigma^{(1)}_{M_{<a}}, \ldots, \Sigma^{(k)}_{M_{<a}}). && \text{(the input of } Q^{(1)}, \ldots, Q^{(k)} \text{ on hyperedges in } M_{<a}\text{)}
\end{aligned}$$

Notice that this way,

$$\Gamma^*_{<a} = \Sigma^{(1)}_{\ell_1,M_{<a}} \circ \cdots \circ \Sigma^{(k)}_{\ell_k,M_{<a}}.$$

The following claim allows us to "over condition" on $\Sigma_{*,M_{<a}}$ instead of $\Gamma^*_{<a}$ in the RHS of Eq (9). The purpose of this step is to isolate the dependence on L which is needed for our subsequent proofs (as $\Gamma^*_{<a}$ depends on L but $\Sigma_{*,M_{<a}}$ does not).

Claim 5.2. *We have:*

$$\mathop{\mathbb{E}}_{\Pi,L,M} \mathop{\mathbb{E}}_{\Gamma^*_{<a}|\Pi,L,M} \|(\gamma_a^\star \mid \Pi, L, M, \Gamma^*_{<a}) - \mathcal{U}_{S_b}\|_{\text{tvd}} \leqslant \mathop{\mathbb{E}}_{\Pi,L,M,\Sigma_{*,M_{<a}}} \|(\gamma_a^\star \mid \Pi, L, M, \Sigma_{*,M_{<a}}) - \mathcal{U}_{S_b}\|_{\text{tvd}}.$$

Proof. For any choice of Π, L, M and $\Gamma^*_{<a}$, we have,

$$\begin{aligned}
\|(\gamma_a^\star \mid \Pi, L, M, \Gamma^*_{<a}) - \mathcal{U}_{S_b}\|_{\text{tvd}} &\leqslant \mathop{\mathbb{E}}_{\Sigma_{*,M_{<a}}|\Pi,L,M,\Gamma^*_{<a}} \|(\gamma_a^\star \mid \Pi, L, M, \Gamma^*_{<a}, \Sigma_{*,M_{<a}}) - \mathcal{U}_{S_b}\|_{\text{tvd}}\\
&\qquad \text{(by the over conditioning property of TVD in Fact A.10)}\\
&= \mathop{\mathbb{E}}_{\Sigma_{*,M_{<a}}|\Pi,L,M,\Gamma^*_{<a}} \|(\gamma_a^\star \mid \Pi, L, M, \Sigma_{*,M_{<a}}) - \mathcal{U}_{S_b}\|_{\text{tvd}}.\\
&\qquad \text{(as } \Sigma_{*,M_{<a}} \text{ and } L \text{ together deterministically fix } \Gamma^*_{<a}\text{)}
\end{aligned}$$

Using this in the LHS of the claim gives us,

$$\begin{aligned}
\mathop{\mathbb{E}}_{\Pi,L,M} \mathop{\mathbb{E}}_{\Gamma^*_{<a}|\Pi,L,M} \|(\gamma_a^\star \mid \Pi, L, M, \Gamma^*_{<a}) - \mathcal{U}_{S_b}\|_{\text{tvd}} &= \mathop{\mathbb{E}}_{\Pi,L,M,\Gamma^*_{<a}} \|(\gamma_a^\star \mid \Pi, L, M, \Gamma^*_{<a}) - \mathcal{U}_{S_b}\|_{\text{tvd}}\\
&\leqslant \mathop{\mathbb{E}}_{\Pi,L,M,\Gamma^*_{<a},\Sigma_{*,M_{<a}}} \|(\gamma_a^\star \mid \Pi, L, M, \Sigma_{*,M_{<a}}) - \mathcal{U}_{S_b}\|_{\text{tvd}}\\
&\qquad \text{(by the above inequality)}\\
&= \mathop{\mathbb{E}}_{\Pi,L,M,\Sigma_{*,M_{<a}}} \|(\gamma_a^\star \mid \Pi, L, M, \Sigma_{*,M_{<a}}) - \mathcal{U}_{S_b}\|_{\text{tvd}},\\
&\qquad \text{(again, since } \Sigma_{*,M_{<a}} \text{ and } L \text{ together deterministically fix } \Gamma^*_{<a}\text{)}
\end{aligned}$$

concluding the proof. ∎

The main distribution. For any choice of $M_{<a}$ and $\Sigma_{*,M_{<a}}$, we define,

Distribution $\mu = \mu_{M_{<a},\Sigma_{*,M_{<a}}}$: Joint distribution of variables in **Multi-HPH** conditioned on this particular choice of $M_{<a}$ and $\Sigma_{*,M_{<a}}$.

For simplicity of notation, and to avoid clutter, we denote the variables in this distribution, for $i \in [k]$ and $j \in [t]$ as follows:

$$
\begin{aligned}
L &:= (\ell_1, \ldots, \ell_k) && \text{(the same input } L \text{ of the referee as before)}\\
R_i &:= [r] - \{M_{1,i}, \ldots, M_{a-1,i}\} && \text{(the set of vertices in layer } i \text{ not matched by } M_{<a})\\
M_a &:= e := (v_1, \ldots, v_k) && \text{(the hyperedge } e = M_a \text{ with vertex } v_i \text{ chosen from } R_i)\\
\bar{\Sigma}^{(i)} &:= \Sigma^{(i)}_{*,R_i} && \text{(the remaining input of player } Q^{(i)} \text{ not fixed by } \Sigma_{*,M_{<a}})\\
\bar{\Sigma} &:= (\bar{\Sigma}^{(1)}, \ldots, \bar{\Sigma}^{(k)}) && \text{(the remaining input of all players } Q^{(1)}, \ldots, Q^{(k)})\\
\gamma^\star &:= \gamma^\star_a := \bar{\sigma}^{(1)}_{\ell_1,v_1} \circ \cdots \circ \bar{\sigma}^{(k)}_{\ell_k,v_k} && \text{(the permutation in } S_b \text{ we are interested in)}\\
\bar{\Pi} &:= (\bar{\Pi}_1, \ldots, \bar{\Pi}_k). && \text{(the messages of players conditioned on } M_{<a} \text{ and } \Sigma_{*,M_{<a}})
\end{aligned}
$$

We list several useful properties of this distribution in the following claim.

Claim 5.3. *For any choice of $M_{<a}, \Sigma_{*,M_{<a}}$, let $\mu := \mu_{M_{<a},\Sigma_{*,M_{<a}}}$. We have,*

(i) $\mu(L)$ is uniform over $[t]^k$;

(ii) R_i is $|R_i| = r - (a-1) \geqslant r/2$ and we can use $r_a := r - (a-1)$ to denote the size of every R_i;

(iii) $\mu(e = (v_1, \ldots, v_k))$ picks each $v_i \in R_i$ independently and uniformly at random;

(iv) $\mu(\bar{\Sigma})$ is uniform distribution over its support $(S_b)^{[t]\times R_1} \times \cdots \times (S_b)^{[t]\times R_k}$;

(v) We have the following independence properties:

$$\mu(\bar{\Sigma}, \bar{\Pi}, \mathsf{L}, \mathsf{e}) = \mu(\bar{\Sigma}, \bar{\Pi}) \times \prod_{i=1}^{k} \mu(\boldsymbol{\ell}_i) \times \prod_{i=1}^{k} \mu(\mathsf{v}_i).$$

Proof. In this proof, all random variables are with respect to the original distribution in **Multi-HPH** unless explicitly indexed by μ.

Proof of Item *(i)*. We have $\mu(\mathsf{L}) = (\mathsf{L} \mid M_{<a}, \Sigma_{*,M_{<a}})$. In **Multi-HPH**, L is chosen independent of M, Σ and is uniform over $[t]^k$, thus $\mu(L)$ is also uniform over $[t]^k$.

Proof of Item *(ii)*. Each hyperedge in $M_{<a}$ uses one vertex in each layer of the graph, thus R_i has $a-1$ matched vertices by $M_{<a}$ and $(r-(a-1))$ unmatched vertices. Moreover, since $a \leqslant r/2$, we get that $r_a = (r-(a-1)) \geqslant r/2$ as well.

Proof of Item *(iii)*. We have $\mu(\mathsf{e}) = (\mathsf{M}_a \mid M_{<a}, \Sigma_{*,M_{<a}})$. In **Multi-HPH**, M_a is chosen independent of Σ and is uniform over all hyperedges of the k-layered graph on $[r]^k$ that are not matched by $M_{<a}$. Given number of unmatched vertices in each layer is exactly $r-(a-1)$ by the previous part, such a hyperedge can be chosen by picking one vertex uniformly at random from unmatched vertices of each level, i.e., from R_i in layer i.

Proof of Item (iv). We have $\mu(\bar{\Sigma}) = (\Sigma \mid M_{<a}, \Sigma_{*,M_{<a}})$. In **Multi-HPH**, every permutation in S_b of Σ is chosen independently of the rest (and independent of M). Thus, after conditioning on $\Sigma_{*,M_{<a}}$, remaining coordinates, i.e., matrices $\Sigma^{(i)}_{*,R_i} \in (S_b)^{[t]\times R_i}$ for $i \in [k]$, are chosen independently.

Proof of Item (v). By the chain rule,

$$(\Pi, \Sigma, \mathsf{L}, \mathsf{M}_a \mid M_{<a}, \Sigma_{*,M_{<a}}) = (\Pi, \Sigma \mid M_{<a}, \Sigma_{*,M_{<a}}) \times (\mathsf{L}, \mathsf{M}_a \mid \Pi, \Sigma, M_{<a}, \Sigma_{*,M_{<a}}).$$

We prove that

$$(\mathsf{L}, \mathsf{M}_a \perp \Pi, \Sigma \mid M_{<a}, \Sigma_{*,M_{<a}}) \quad \equiv \quad \mathbb{I}(\mathsf{L}, \mathsf{M}_a ; \Pi, \Sigma \mid M_{<a}, \Sigma_{*,M_{<a}}) = 0,$$

where the equivalence is by Fact A.1-(2). We have,

$$\begin{aligned}
\mathbb{I}(\mathsf{L}, \mathsf{M}_a ; \Pi, \Sigma \mid M_{<a}, \Sigma_{*,M_{<a}}) &= \mathbb{I}(\mathsf{L}, \mathsf{M}_a ; \Sigma \mid M_{<a}, \Sigma_{*,M_{<a}}) + \mathbb{I}(\mathsf{L}, \mathsf{M}_a ; \Pi \mid \Sigma, M_{<a}, \Sigma_{*,M_{<a}}) \\
&\qquad \text{(by chain rule of mutual information in Fact A.1-(6))} \\
&= \mathbb{I}(\mathsf{L}, \mathsf{M}_a ; \Sigma \mid M_{<a}, \Sigma_{*,M_{<a}}) \\
&\text{(as } \Pi \text{ is fixed by } \Sigma \mid M_{<a}, \Sigma_{*,M_{<a}} \text{ and thus the second term is zero by Fact A.1-(2))} \\
&= 0. \qquad \text{(by Fact A.1-(2) as } \Sigma \text{ is chosen independent of } \mathsf{L}, \mathsf{M})
\end{aligned}$$

This implies that for every choice of $M_{<a}, \Sigma_{*,M_{<a}}$,

$$\begin{aligned}
\mu(\bar{\Sigma}, \bar{\Pi}, \mathsf{L}, \mathsf{e}) &= (\Pi, \Sigma, \mathsf{L}, \mathsf{M}_a \mid M_{<a}, \Sigma_{*,M_{<a}}) = (\Pi, \Sigma \mid M_{<a}, \Sigma_{*,\mathsf{M}_{<a}}) \times (\mathsf{L}, \mathsf{M}_a \mid M_{<a}, \Sigma_{*,M_{<a}}) \\
&= \mu(\bar{\Pi}, \bar{\Sigma}) \times \mu(\mathsf{L}, \mathsf{e}) = \mu(\bar{\Pi}, \bar{\Sigma}) \times \mu(\mathsf{L}) \times \mu(\mathsf{e} \mid \mathsf{L}).
\end{aligned}$$

by the definition of variables $(\bar{\Sigma}, \bar{\Pi}, \mathsf{L}, \mathsf{e})$.

By Item (i), we have $\mu(\mathsf{L}) = \prod_{i=1}^{k} \mu(\boldsymbol{\ell}_i)$. We also have $\mu(\mathsf{e} \mid \mathsf{L}) = (\mathsf{M}_a \mid \mathsf{L}, M_{<a}, \Sigma_{*,M_{<a}})$ which is the same as $\mu(e)$ as M_a is chosen independent of L. Thus, by Item (iii), we also have $\mu(\mathsf{e} \mid \mathsf{L}) = \mu(\mathsf{e}) = \prod_{i=1}^{k} \mu(\mathsf{v}_i)$, concluding the proof. ∎

We can now write the RHS of Claim 5.2 in terms of the distribution $\mu = \mu_{M_{<a},\Sigma_{*,M_{<a}}}$ as follows:

$$\begin{aligned}
\text{RHS of Claim 5.2} &= \mathop{\mathbb{E}}_{M_{<a},\Sigma_{*,M_{<a}}} \mathop{\mathbb{E}}_{\Pi,L,M_{\geqslant a}\sim\mu} \|\mu(\boldsymbol{\gamma}^\star_a \mid \Pi, L, M_{\geqslant a}) - \mathcal{U}_{S_b}\|_{\text{tvd}} \\
&\qquad \text{(by the definition of } \mu \text{ as conditioning on } M_{<a}, \Sigma_{*,M_{<a}}) \\
&= \mathop{\mathbb{E}}_{M_{<a},\Sigma_{*,M_{<a}}} \mathop{\mathbb{E}}_{\Pi,L,e\sim\mu} \|\mu(\boldsymbol{\gamma}^\star \mid \Pi, L, e) - \mathcal{U}_{S_b}\|_{\text{tvd}} \\
&\text{(by Claim 5.3-Item (v), } \gamma^\star = \gamma^\star_a \text{ is only a function of } e = M_a \text{ among } M_{\geqslant a}) \\
&= \mathop{\mathbb{E}}_{M_{<a},\Sigma_{*,M_{<a}}} \mathop{\mathbb{E}}_{\bar{\Pi}\sim\mu} \mathop{\mathbb{E}}_{L,e=(v_1,\ldots,v_k)\sim\mu} \|\mu(\bar{\boldsymbol{\sigma}}^{(1)}_{\ell_1,v_1} \circ \cdots \circ \bar{\boldsymbol{\sigma}}^{(k)}_{\ell_k,v_k} \mid \Pi) - \mathcal{U}_{S_b}\|_{\text{tvd}}.
\end{aligned}$$

(by Claim 5.3-Item (v), $\bar{\Pi} \perp L, e$ in μ and $\bar{\Sigma}$ is only a function of $\bar{\Pi}$ after fixing (ℓ_i, v_i) for $i \in [k]$)

By plugging in these bounds for the RHS of Claim 5.2 in Eq (9) and then in Eq (8), we get that

$$\Pr(\pi \text{ succeeds}) \leqslant \frac{1}{2} + \frac{r}{2} \cdot \max_{\substack{M_{<a},\Sigma_{*,M_{<a}} \\ \mu=\mu_{M_{<a},\Sigma_{*,M_{<a}}}}} \left[\mathop{\mathbb{E}}_{\bar{\Pi}\sim\mu} \mathop{\mathbb{E}}_{\substack{\ell_1,\ldots,\ell_k \\ v_1,\ldots,v_k\sim\mu}} \|\mu(\bar{\boldsymbol{\sigma}}^{(1)}_{\ell_1,v_1} \circ \cdots \circ \bar{\boldsymbol{\sigma}}^{(k)}_{\ell_k,v_k} \mid \Pi) - \mathcal{U}_{S_b}\|_{\text{tvd}} \right]. \quad (10)$$

This concludes the basic setup we have for proving the lower bound for **Multi-HPH**. The remaining subsections prove an upper bound for the RHS of Eq (10), which is the heart of the proof.

Conditioning Step: For the remainder of this section, we fix an arbitrary choice of $M_{<a}, \Sigma_{*,M_{<a}}$ and $\mu = \mu_{M_{<a},\Sigma_{*,M_{<a}}}$ and all random variables are with respect to μ unless specified otherwise.

Problem 2. *It is worth explicitly identifying the problem we need to solve when bounding the RHS of Eq (10) for the distribution μ. The problem we are interested in is as follows:*

(*i*) *We have k players $Q^{(1)}, \ldots, Q^{(k)}$ each getting a matrix $\bar{\Sigma}^{(i)} \in (S_b)^{t \times r_a}$, with each entry being a permutation in S_b chosen uniformly at random and independently.*

(*ii*) *We also have a referee that picks k random and independently chosen entries of this matrix, namely, $(\ell_i, v_i) \in [t] \times [r_a]$ for $i \in [k]$. We refer to $(\bar{\sigma}^{(1)}_{\ell_1,v_1}, \ldots, \bar{\sigma}^{(k)}_{\ell_k,v_k})$ as the **hidden permutations** of the players. We also define the **target permutation** $\gamma^\star$ as:*

$$\gamma^\star := \bar{\sigma}^{(1)}_{\ell_1,v_1} \circ \cdots \circ \bar{\sigma}^{(k)}_{\ell_k,v_k}.$$

We are interested in communication protocols for this problem that follow the same rules as the ones for **Multi-HPH**, *with the different goal of changing the distribution of the target permutation from its original uniform distribution as much as possible.*

5.2 Part Two: KL-Divergence of Individual Hidden Permutations

In this subsection, we prove that not many players are able to reveal much information about their hidden permutation, measured as the KL-divergence of their hidden permutation conditioned on the transcript from the uniform distributions. This is proven using a simple *direct-product* style argument to show that not only each player is unable to reveal much about their hidden permutation, but in fact this is true *simultaneously* for most players.

To continue, we need the following definition.

Definition 5.4. Set the parameter

$$\theta := \left(\frac{4s + 4k \cdot \log(t \cdot r_a)}{t \cdot r_a} \right)^{1/2}. \tag{11}$$

(Notice that by the bound we have on s in Theorem 2, $\theta < 1/100$).

For any $i \in [k]$ and $(\ell, v) \in [t] \times [r_a]$, we say that a transcript $\bar{\Pi}$ is **informative** for player $Q^{(i)}$ on the index (ℓ, v) iff:

$$\mathbb{D}(\mu(\bar{\boldsymbol{\sigma}}^{(i)}_{\ell,v} \mid \bar{\Pi}) \mid\mid \mathcal{U}_{S_b}) > \theta.$$

Moreover, for a transcript $\bar{\Pi}$ and input $L = (\ell_1, \ldots, \ell_k)$, $e = (v_1, \ldots, v_k)$ to the referee, we define the set of **informed indices** as:

$$\textsc{Info} = \textsc{Info}(\bar{\Pi}, L, e) := \left\{ i \in [k] \mid \Pi \text{ is informative for player } Q^{(i)} \text{ on } (\ell_i, v_i) \right\}.$$

Informally, this definition captures which permutations the referee has a lot of information about. Conditioned on any transcript and input indices to the referee, if the distribution of $\bar{\sigma}^{(i)}_{\ell_i,v_i}$ differs from $\mathcal{U}_{S_b}$ by θ in KL-Divergence, the referee has a better chance of guessing $\bar{\sigma}^{(i)}_{\ell_i,v_i}$, which in turn helps with reducing the uncertainty on the target permutation $\gamma^\star$.

The following lemma shows that size of $\textsc{Info}$ cannot be too large with high probability.

Lemma 5.5 ("Informed indices cannot be too many")**.**

$$\Pr_{(\bar{\Pi},L,e)}\Big(|\textsc{Info}(\bar{\Pi},L,e)|>k/2\Big)\leqslant 2\cdot(16\cdot\theta)^{k/4}.$$

Proof. The proof consists of two parts. The first part is to show that with high probability over the choice of $\bar{\Pi}$, the KL-divergence of $\bar{\Sigma}^{(i)}$ conditioned on the transcript from its original distribution for "most" indices $i\in[k]$ is sufficiently "low". The second part then shows that only a small fraction of these low KL-divergence indices can become informative, which concludes the proof.

Part I: KL-divergence of $\bar{\Sigma}^{(i)}$ is "low" for "most" $i\in[k]$. For any transcript $\bar{\Pi}$, define:

$$L(\bar{\Pi}):=\left\{i\in[k]\;\middle|\;\mathbb{D}(\bar{\boldsymbol{\Sigma}}^{(i)}\mid\bar{\boldsymbol{\Pi}}=\bar{\Pi}\;||\;\bar{\boldsymbol{\Sigma}}^{(i)})\leqslant 4s+4k\cdot\log(1/\theta)\right\},$$

namely, the indices in $[k]$ with low KL-divergence conditioned on $\bar{\Pi}$ from their original distribution. We are going to prove that $L(\bar{\Pi})$ is going to be large quite likely.

Claim 5.6. $\Pr_{\bar{\Pi}}\left(\left|L(\bar{\Pi})\right|\leqslant 3k/4\right)\leqslant\theta^k$.

Proof. Let S be the set of all transcripts $\bar{\Pi}$ such that,

$$\mathbb{D}(\bar{\boldsymbol{\Sigma}}\mid\bar{\boldsymbol{\Pi}}=\bar{\Pi}\;||\;\bar{\boldsymbol{\Sigma}})>s+k\cdot\log(1/\theta).$$

Note that since $\bar{\Pi}$ is a deterministic function of $\bar{\boldsymbol{\Sigma}}$, by Fact A.6 (the moreover part), we have

$$\mathbb{D}(\bar{\boldsymbol{\Sigma}}\mid\bar{\boldsymbol{\Pi}}=\bar{\Pi}\;||\;\bar{\boldsymbol{\Sigma}})=\log\Big(\frac{1}{\Pr(\bar{\boldsymbol{\Pi}}=\bar{\Pi})}\Big).$$

Thus, for each $\bar{\Pi}\in S$, we have,

$$\Pr\left(\bar{\boldsymbol{\Pi}}=\bar{\Pi}\right)\leqslant 2^{-s-k\cdot\log(1/\theta)}.$$

At the same time, since there are at most 2^s choices for $\bar{\Pi}$, by union bound, we have,

$$\Pr_{\bar{\Pi}}\left(\bar{\Pi}\in S\right)\leqslant\sum_{\bar{\Pi}\in S}\Pr\left(\bar{\boldsymbol{\Pi}}=\bar{\Pi}\right)\leqslant 2^s\cdot 2^{-s-k\cdot\log(1/\theta)}=\theta^k.$$

In the following, we condition on $\bar{\Pi}\notin S$ which happens with probability $1-\theta^k$.

We now have,

$$\begin{aligned}s+k\cdot\log(1/\theta)&\geqslant\mathbb{D}(\bar{\boldsymbol{\Sigma}}\mid\bar{\boldsymbol{\Pi}}=\bar{\Pi}\;||\;\bar{\boldsymbol{\Sigma}})\qquad\text{(by the definition of }\bar{\Pi}\text{ being not informative)}\\&\geqslant\sum_{i=1}^{k}\mathbb{D}(\bar{\boldsymbol{\Sigma}}^{(i)}\mid\bar{\boldsymbol{\Pi}}=\bar{\Pi}\;||\;\bar{\boldsymbol{\Sigma}}^{(i)}).\end{aligned}$$

($\bar{\boldsymbol{\Sigma}}$ is a product distribution (by Claim 5.3-Item *(iv)*) so we can apply Fact A.5 (moreover part))

A simple Markov bound implies that at most one-fourth of the indices can have KL-divergence more than $4s+4k\cdot\log(1/\theta)$. Thus, for any $\bar{\Pi}\notin S$, $L(\bar{\Pi})$ is of size at least $3k/4$. The bound of θ^k above on the probability of $\bar{\Pi}$ being in S concludes the proof. ∎

In the following, we fix any choice of $\bar{\Pi}$ with $L(\bar{\Pi})$ having size at least $3k/4$, which by Claim 5.6 happens with probability at least $1-\theta^k$.

Part II: "Few" indices in $L(\bar{\Pi})$ can be informative. We bound the size of informative indices in $L(\bar{\Pi})$ as follows.

Claim 5.7. $\Pr_{(L,e)|\bar{\Pi}}\left(\geqslant k/4 \text{ indices in } L(\bar{\Pi}) \text{ are informative}\right) \leqslant (16\theta)^{k/4}$.

Proof. For any $i \in L(\bar{\Pi})$,

$$\mathbb{D}(\bar{\Sigma}^{(i)} \mid \bar{\Pi} = \bar{\Pi} \mid\mid \bar{\Sigma}^{(i)}) \geqslant \sum_{\ell=1}^{t} \sum_{v \in R_i} \mathbb{D}(\bar{\boldsymbol{\sigma}}^{(i)}_{\ell,v} \mid \bar{\Pi} = \bar{\Pi} \mid\mid \bar{\boldsymbol{\sigma}}^{(i)}_{\ell,v}).$$

($\bar{\Sigma}$ is a product distribution (by Claim 5.3-Item (iv)) so we can apply Fact A.5 (moreover part))

Thus, by the KL-divergence bound of the previous part for indices $i \in L(\bar{\Pi})$,

$$\mathop{\mathbb{E}}_{(\ell_i,v_i)|\bar{\Pi}} \mathbb{D}(\bar{\boldsymbol{\sigma}}^{(i)}_{\ell_i,v_i} \mid \bar{\Pi} = \bar{\Pi} \mid\mid \bar{\boldsymbol{\sigma}}^{(i)}_{\ell_i,v_i}) \leqslant \frac{4s + 4k \cdot \log(1/\theta)}{t \cdot r_a} \leqslant \frac{4s + 4k \cdot \log(tr_a)}{t \cdot r_a} = \theta^2,$$

where the first inequality is because $(\ell, v) \mid \bar{\Pi}$ is uniformly distributed over $[t] \times R_i$ (by Claim 5.3-Item (v)), and the second inequality and subsequent equality hold by the choice of θ.

We can now apply Markov bound and obtain that for every $i \in L(\bar{\Pi})$,

$$\Pr_{(\ell_i,v_i)|\bar{\Pi}}\left(\mathbb{D}(\bar{\boldsymbol{\sigma}}^{(i)}_{\ell_i,v_i} \mid \bar{\Pi} = \bar{\Pi} \mid\mid \bar{\boldsymbol{\sigma}}^{(i)}_{\ell_i,v_i}) > \theta\right) \leqslant \theta.$$

Since $\bar{\boldsymbol{\sigma}}^{(i)}_{\ell_i,v_i}$ is distributed as $\mathcal{U}_{S_b}$ (by Claim 5.3-Item (iv)), this implies that for every $i \in L(\bar{\Pi})$,

$$\Pr_{(\ell_i,v_i)|\bar{\Pi}}\left(i \text{ is informative}\right) \leqslant \theta.$$

In addition, since the choice of $(\ell_i, v_i) \mid \bar{\Pi}$ for all $i \in L(\bar{\Pi})$ (generally in $[k]$) are independent of each other (by Claim 5.3-Item (v)), we have that,

$$\begin{aligned}\Pr_{(L,e)|\bar{\Pi}}\left(\geqslant k/4 \text{ indices in } L(\bar{\Pi}) \text{ are informative}\right) &\leqslant \sum_{\substack{S \subseteq L(\bar{\Pi}) \\ |S| = k/4}} \prod_{i \in S} \Pr_{(\ell_i,v_i)|\bar{\Pi}}\left(i \text{ is informative}\right) \\ &\leqslant 2^k \cdot \theta^{k/4} = (16\theta)^{k/4}. \qquad \blacksquare\end{aligned}$$

Conclusion. We can now conclude the proof of Lemma 5.5. By Claim 5.6, with probability $1-\theta^k$ over the choice of $\bar{\Pi}$, at least $3k/4$ indices are in $L(\bar{\Pi})$ (and so we can have at most $k/4$ informative indices outside $L(\bar{\Pi})$). For any $\bar{\Pi}$ with $\left|L(\bar{\Pi})\right| \geqslant 3k/4$, by Claim 5.7, at most $k/4$ indices $i \in L(\bar{\Pi})$ can be informative for $(\bar{\Pi}, L, e)$. Thus, by union bound,

$$\Pr_{(\bar{\Pi},L,e)}\left(|\textsc{Info}(\bar{\Pi}, L, e)| > k/2\right) \leqslant \theta^k + (16 \cdot \theta)^{k/4} \leqslant 2 \cdot (16 \cdot \theta)^{k/4}.$$

This finalizes the proof of Lemma 5.5. $\blacksquare$

Conditioning Step: For the rest of the proof, we condition on any choice of $(\bar{\Pi}, L, e)$ such that

$$\textsc{Info} := \textsc{Info}(\bar{\Pi}, L, e) \leqslant k/2, \tag{12}$$

and we know that by Lemma 5.5, this event happens with high probability.

5.3 Part Three: Total Variation Distance of the Target Permutation

At this point of the argument, we have already fixed the choice of $(\bar{\Pi}, L, e)$ which, by Eq (12), gives us that for many indices $i \in [k]$, the distribution of the hidden permutation $\bar{\sigma}^{(i)}_{\ell_i,v_i}$ is close to uniform in the KL-divergence. Recall that the target distribution $\gamma^\star$ is

$$\gamma^\star = \bar{\sigma}^{(1)}_{\ell_1,v_1} \circ \cdots \circ \bar{\sigma}^{(k)}_{\ell_k,v_k}.$$

Our goal is to show that this concatenation is going to make the distribution of $\gamma^\star$ exponentially closer to uniform (albeit in another measure of distance, not KL-divergence). This is proven in the following three steps:

- **Step I:** We first prove that even conditioned on $(\bar{\Pi}, L, e)$, the distribution of hidden permutations are independent of each other. This is an application of the rectangle property of protocols.
- **Step II:** We then show that for many (but not all) of the hidden permutations, the KL-divergence bounds can be translated to bounds on the ℓ_2-distance of the distribution from uniform. A crucial tool we use here is an strengthened Pinsker's inequality, due to [CK18], that bounds a mixture of ℓ_1- and ℓ_2-distance between distributions via KL-divergence. The main part of the argument here is to handle the ℓ_1-distance bounds, and postpone the ℓ_2-distances to the next step.
- **Step III:** Finally, we use the Fourier analytic tools of Appendix B combined with the ℓ_2-distance bounds of the previous step, to bound the ℓ_2-distance of the distribution of $\gamma^\star$ from uniform, and get the desired bound on the total variation distance as well.

5.3.1 Step I: Conditional Independence of Inputs Even After Communication

We prove that players' inputs remain independent even conditioned on the messages and input of the referee. The proof is a standard application of the rectangle property of communication protocols extended to the multi-party setting and two specific parts in the inputs. But, despite its simplicity, this lemma plays a crucial rule in our arguments in the last step.

Lemma 5.8. *For any transcript $\bar{\Pi}$ and input $L = (\ell_1, \ldots, \ell_k), e = (v_1, \ldots, v_k)$ to the referee in μ,*

$$\mu(\bar{\boldsymbol{\sigma}}^{(1)}_{\ell_1,v_1}, \ldots, \bar{\boldsymbol{\sigma}}^{(k)}_{\ell_k,v_k} \mid \bar{\Pi}, L, e) = \prod_{i=1}^{k} \mu(\bar{\boldsymbol{\sigma}}^{(i)}_{\ell_i,v_i} \mid \bar{\Pi});$$

Proof. By the chain rule of probabilities,

$$\mu(\bar{\boldsymbol{\sigma}}^{(1)}_{\ell_1,v_1}, \ldots, \bar{\boldsymbol{\sigma}}^{(k)}_{\ell_k,v_k} \mid \bar{\Pi}, L, e) = \prod_{i=1}^{k} \mu(\bar{\boldsymbol{\sigma}}^{(i)}_{\ell_i,v_i} \mid \bar{\Pi}, \bar{\sigma}^{(<i)}_{L_{<i},e_{<i}}, L, e) = \prod_{i=1}^{k} \mu(\bar{\boldsymbol{\sigma}}^{(i)}_{\ell_i,v_i} \mid \bar{\Pi}, \bar{\sigma}^{(<i)}_{L_{<i},e_{<i}}),$$

where the final equality is by Claim 5.3-Item (v) because (L, e) are independent of $(\bar{\Sigma}, \bar{\Pi})$. Thus, to prove the lemma, we only need to prove that for every $i \in [k]$,

$$\left(\bar{\boldsymbol{\sigma}}^{(i)}_{\ell_i,v_i} \perp \bar{\boldsymbol{\sigma}}^{(<i)}_{L_{<i},e_{<i}} \mid \bar{\Pi}\right) \quad \equiv \quad \mathbb{I}(\bar{\boldsymbol{\sigma}}^{(i)}_{\ell_i,v_i}; \bar{\boldsymbol{\sigma}}^{(<i)}_{L_{<i},e_{<i}} \mid \bar{\Pi}) = 0,$$

where the equivalence is by Fact A.1-(2).

We have,

$$\mathbb{I}(\bar{\boldsymbol{\sigma}}^{(i)}_{\ell_i,v_i}; \bar{\boldsymbol{\sigma}}^{(<i)}_{L_{<i},e_{<i}} \mid \bar{\Pi}) \leqslant \mathbb{I}(\bar{\Sigma}^{(i)}; \bar{\Sigma}^{(-i)} \mid \bar{\Pi})$$

by the data processing inequality (Fact A.1-(7)), as $\bar{\Sigma}^{(i)}$ fixes $\bar{\boldsymbol{\sigma}}^{(i)}_{\ell_i,v_i}$ and $\bar{\Sigma}^{(-i)}$ fixes $\bar{\boldsymbol{\sigma}}^{(<i)}_{L_{<i},e_{<i}}$; notice that here (L, e) are fixed and we are looking at specific indices of $\bar{\boldsymbol{\sigma}}^{(i)}_{\ell_i,v_i}$ of $\bar{\Sigma}^{(i)}$ and $\bar{\boldsymbol{\sigma}}^{(<i)}_{L_{<i},e_{<i}}$ of $\bar{\Sigma}^{(-i)}$, and as such we can indeed apply the data processing inequality.

With a slight abuse of notation, we denote $\bar{\Pi} = \bar{\Pi}(1), \ldots, \bar{\Pi}(s)$ where for all $j \in [s]$, $\bar{\Pi}(j)$ denotes the j-th bit communicated by the players. We claim that for every $j \in [s]$,

$$\mathbb{I}(\bar{\Sigma}^{(i)}; \bar{\Sigma}^{(-i)} \mid \bar{\Pi}(1), \ldots, \bar{\Pi}(j)) \leqslant \mathbb{I}(\bar{\Sigma}^{(i)}; \bar{\Sigma}^{(-i)} \mid \bar{\Pi}(1), \ldots, \bar{\Pi}(j-1));$$

This is because:

- if the j-th bit of the protocol is sent by player $Q^{(i)}$, then it is a deterministic function of $\bar{\Sigma}^{(i)}$ and $\bar{\Pi}(1), \ldots, \bar{\Pi}(j-1)$ and thus

  $$\bar{\Pi}(j) \perp \bar{\Sigma}^{(-i)} \mid \bar{\Sigma}^{(i)}, \bar{\Pi}(1), \ldots, \bar{\Pi}(j-1);$$

  hence, the inequality holds by Proposition A.3.

- if the j-th bit of the protocol is sent by any player other than $Q^{(i)}$, then it is a deterministic function of $\bar{\Sigma}^{(-i)}$ and $\bar{\Pi}(1), \ldots, \bar{\Pi}(j-1)$ and thus

  $$\bar{\Pi}(j) \perp \bar{\Sigma}^{(i)} \mid \bar{\Sigma}^{(-i)}, \bar{\Pi}(1), \ldots, \bar{\Pi}(j-1);$$

  hence, again, the inequality holds by Proposition A.3.

Applying this inequality repeatedly then gives us

$$\mathbb{I}(\bar{\Sigma}^{(i)}; \bar{\Sigma}^{(-i)} \mid \bar{\Pi}(1), \ldots, \bar{\Pi}(j)) \leqslant \mathbb{I}(\bar{\Sigma}^{(i)}; \bar{\Sigma}^{(-i)}) = 0,$$

where the second equality holds by Fact A.1-(2) because of the independence of the parameters (as shown in Claim 5.3-Item *(iv)*). This concludes the proof of the lemma. ∎

5.3.2 Step II: From KL-Divergence to "Strong" ℓ_2-Bounds

Recall that we already fixed a choice of $(\bar{\Pi}, L, e)$ that guarantees $\textsc{Info} = \textsc{Info}(\bar{\Pi}, L, e)$ has size at most $k/2$ by Eq (12). We further define the following two sets for every $i \in [k]$:

$$A_i := \left\{\sigma \in S_b \mid \mu(\bar{\boldsymbol{\sigma}}^{(i)}_{\ell_i,v_i} = \sigma \mid \bar{\Pi}) > 2/b!\right\};$$
$$B_i := S_b \setminus A_i.$$

In words, A_i is the set of those permutations in S_b whose probability mass under $\bar{\boldsymbol{\sigma}}^{(i)}_{\ell_i,v_i} \mid \bar{\Pi}$ is "much higher" than that of uniform distribution (more precisely, more than twice), and B_i collects the remaining permutations. We also define two events:

- **Event** $\mathcal{E}_A(i)$: the sampled input $\bar{\boldsymbol{\sigma}}^{(i)}_{\ell_i,v_i}$ belongs to A_i;
- **Event** $\mathcal{E}_B(i)$: the sampled input $\bar{\boldsymbol{\sigma}}^{(i)}_{\ell_i,v_i}$ belongs to B_i.

Note that these two events are complement of each other and only depend on the choice of $\bar{\sigma}^{(i)}_{\ell_i,v_i}$ at this point. Finally, we define the set of **good** indices $i \in [k]$ as:

$$\textsc{Good} := \{i \in [k] \mid i \text{ is } \underline{\text{not}} \text{ in } \textsc{Info} \text{ and } \mathcal{E}_B(i) \text{ happens}\}.$$

The reason we consider indices in Good as "good" is because we can translate our KL-divergence bound on $\bar{\boldsymbol{\sigma}}^{(i)}_{\ell_i,v_i}$ for $i \in \textsc{Good}$ (even conditioned on them being in good) into a "strong" bound on their ℓ_2-distance from the uniform distribution (this is formalized in Lemma 5.10).

We start by showing that $\mathcal{E}_B$ happens frequently for indices not in Info.

Claim 5.9. *For any $i \notin$ Info:*

$$\Pr\left(\mathcal{E}_B(i) \mid \bar{\Pi}\right) \geqslant 1 - \sqrt{8\theta}.$$

Proof. By the definition of i not being in Info and by Pinsker's inequality of Fact A.12, we have,

$$\|\mu(\bar{\boldsymbol{\sigma}}^{(i)}_{\ell_i,v_i} \mid \bar{\Pi}) - \mathcal{U}_{S_b}\|_{\text{tvd}} \leqslant \sqrt{\frac{1}{2} \cdot \mathbb{D}(\mu(\bar{\boldsymbol{\sigma}}^{(i)}_{\ell_i,v_i} \mid \bar{\Pi}) \mid\mid \mathcal{U}_{S_b})} \leqslant \sqrt{\frac{\theta}{2}}.$$

On the other hand,

$$\begin{aligned}\|\mu(\bar{\boldsymbol{\sigma}}^{(i)}_{\ell_i,v_i} \mid \bar{\Pi}) - \mathcal{U}_{S_b}\|_{\text{tvd}} &\geqslant \frac{1}{2} \cdot \sum_{\sigma \in A_i} \left|\mu(\bar{\boldsymbol{\sigma}}^{(i)}_{\ell_i,v_i} = \sigma \mid \bar{\Pi}) - \frac{1}{b!}\right| \\ &\geqslant \frac{1}{4} \cdot \sum_{\sigma \in A_i} \mu(\bar{\boldsymbol{\sigma}}^{(i)}_{\ell_i,v_i} = \sigma \mid \bar{\Pi}),\end{aligned}$$

by the definition of A_i (note that all permutations in A_i have a probability of at least $2/b!$ in our distribution, higher than compared to the uniform distribution). Combining the above two equations gives us:

$$\Pr\left(\mathcal{E}_A(i) \mid \bar{\Pi}\right) = \sum_{\sigma \in A_i} \mu(\bar{\boldsymbol{\sigma}}^{(i)}_{\ell_i,v_i} = \sigma \mid \bar{\Pi}) \leqslant \sqrt{8\theta}.$$

As $\mathcal{E}_B(i)$ is the complement of $\mathcal{E}_A(i)$, we can conclude the proof. ∎

A simple corollary of Claim 5.9 is the following (probabilistic) lower bound on the size of Good:

$$\Pr\left(|\textsc{Good}| < k/4\right) \leqslant \binom{k/2}{k/4} \cdot (8\theta)^{k/8} \leqslant (128\theta)^{k/8}; \tag{13}$$

here, we used the fact that for Good to be of size less than $k/4$, at least $k/4$ indices from the first $k/2$ indices of Info should have $\mathcal{E}_A$ happen for them; we then used Claim 5.9 to bound this probability and combined it with Lemma 5.8 to crucially use the independence of the events $\mathcal{E}_A(i)$ for different values of $i \in [k]$ (the second inequality is by just upper bounding $\binom{k/2}{k/4}$ by $2^{k/2} = 16^{k/8}$).

The following lemma now establishes why the indices in Good are "good" for our purpose: on these indices, the distribution of $\bar{\sigma}^{(i)}_{\ell_i,v_i}$ is "quite" close to uniform in ℓ_2-distance, much closer than a θ^2-bound that follows from directly applying Pinsker's inequality to the KL-divergence bound of indices not in Info; we prove this using the ℓ_2/ℓ_1-version of Pinsker's inequality due to [CK18] mentioned in Proposition A.13.

Lemma 5.10. *For any $i \in$ Good:*

$$\|\mu(\bar{\boldsymbol{\sigma}}^{(i)}_{\ell_i,v_i} \mid \bar{\Pi}, i \in \textsc{Good}) - \mathcal{U}_{S_b}\|_2^2 \leqslant \frac{20\sqrt{\theta}}{b!}.$$

Proof. Firstly, consider an index $i \notin$ Info (but we still do not condition on i being in Good or not). By the definition of Info, we have,

$$\mathbb{D}(\mu(\bar{\boldsymbol{\sigma}}^{(i)}_{\ell_i,v_i} \mid \bar{\Pi}) \mid\mid \mathcal{U}_{S_b}) \leqslant \theta.$$

By applying Proposition A.13 to this KL-divergence bound (taking $A = A_i$ and $B = B_i$ in the proposition), we get

$$\sum_{\sigma \in A_i} \left|\mu(\bar{\boldsymbol{\sigma}}^{(i)}_{\ell_i,v_i} = \sigma \mid \bar{\Pi}) - \frac{1}{b!}\right| + \sum_{\sigma \in B_i} \frac{\left(\mu(\bar{\boldsymbol{\sigma}}^{(i)}_{\ell_i,v_i} = \sigma \mid \bar{\Pi}) - \frac{1}{b!}\right)^2}{\mu(\bar{\boldsymbol{\sigma}}^{(i)}_{\ell_i,v_i} = \sigma \mid \bar{\Pi})} \leqslant \frac{1}{1-\ln 2} \cdot \theta \leqslant 4\theta.$$

By just taking the second term, and since for all $\sigma \in B_i$, $\mu(\bar{\boldsymbol{\sigma}}^{(i)}_{\ell_i,v_i} = \sigma \mid \bar{\Pi}) \leqslant 2/b!$, we get that

$$\sum_{\sigma \in B_i} \left(\mu(\bar{\boldsymbol{\sigma}}^{(i)}_{\ell_i,v_i} = \sigma \mid \bar{\Pi}) - \frac{1}{b!}\right)^2 \leqslant \frac{8\theta}{b!}. \tag{14}$$

We now consider the effect of conditioning on $i \in$ Good as well. Define $\delta_i \in \mathbb{R}$ such that

$$(1+\delta_i) := \mu(i \in \textsc{Good} \mid \bar{\Pi})^{-1} = \Big(\mu(\mathcal{E}_B(i) \mid \bar{\Pi})\Big)^{-1},$$

which by Claim 5.9 and because $\theta < 1/100$ (see Definition 5.4), leads to

$$\delta_i \leqslant \sqrt{18 \cdot \theta}. \tag{15}$$

This gives us, for every $\sigma \in B_i$,

$$\mu(\bar{\boldsymbol{\sigma}}^{(i)}_{\ell_i,v_i} = \sigma \mid \bar{\Pi}, i \in \textsc{Good}) = \frac{\mu(\bar{\boldsymbol{\sigma}}^{(i)}_{\ell_i,v_i} = \sigma \wedge i \in \textsc{Good} \mid \bar{\Pi})}{\mu(i \in \textsc{Good} \mid \bar{\Pi})} = (1+\delta_i) \cdot \mu(\bar{\boldsymbol{\sigma}}^{(i)}_{\ell_i,v_i} = \sigma \mid \bar{\Pi}), \tag{16}$$

where in the last equality, we used the fact that for $\sigma \in B_i$, $\bar{\boldsymbol{\sigma}}^{(i)}_{\ell_i,v_i} = \sigma$ also implies $i \in$ Good (as $\mathcal{E}_B(i)$ happens). We can now calculate the ℓ_2-distance of our desired distribution from $\mathcal{U}_{S_b}$. For the simplicity of exposition in the following calculations, we denote,

$$\nu(\sigma) := \mu(\bar{\boldsymbol{\sigma}}^{(i)}_{\ell_i,v_i} = \sigma \mid \bar{\Pi}).$$

We have,

$$\begin{aligned}
&\|\mu(\bar{\boldsymbol{\sigma}}^{(i)}_{\ell_i,v_i} = \sigma \mid \bar{\Pi}, i \in \textsc{Good}) - \mathcal{U}_{S_b}\|_2^2 \\
&= \sum_{\sigma \in B_i} \left(\mu(\bar{\boldsymbol{\sigma}}^{(i)}_{\ell_i,v_i} = \sigma \mid \bar{\Pi}, i \in \textsc{Good}) - \frac{1}{b!}\right)^2 \\
&\qquad\qquad \text{(conditioning on } i \in \textsc{Good}\text{, implies that only } \sigma \in B_i \text{ have non-zero probability)} \\
&= \sum_{\sigma \in B_i} \left((1+\delta_i) \cdot \nu(\sigma) - \frac{1}{b!}\right)^2 \qquad \text{(by Eq (16) and the definition of } \nu(\sigma) \text{ above)}
\end{aligned}$$

$$
\begin{aligned}
&= \sum_{\sigma\in B_i}\left(\left(\nu(\sigma)-\frac{1}{b!}\right)^2+{\delta_i}^2\cdot\nu(\sigma)^2+2\delta_i\cdot\nu(\sigma)\cdot(\nu(\sigma)-\frac{1}{b!}))\right)\\
&\leqslant \sum_{\sigma\in B_i}\left(\left(\nu(\sigma)-\frac{1}{b!}\right)^2+{\delta_i}^2\cdot\nu(\sigma)^2+2\delta_i\cdot\nu(\sigma)\cdot\frac{1}{b!}\right) && \text{(as } \nu(\sigma)\leqslant 2/b! \text{ since } \sigma\in B_i\text{)}\\
&\leqslant \left(\sum_{\sigma\in B_i}\left(\nu(\sigma)-\frac{1}{b!}\right)^2\right)+\frac{4{\delta_i}^2}{b!}+\frac{4\delta_i}{b!} && \text{(by the bound of } \nu(\sigma)\leqslant 2/b! \text{ for } \sigma\in B_i\text{, and } |B_i|\leqslant b!\text{)}\\
&\leqslant \frac{8\theta+72\theta+\sqrt{288\cdot\theta}}{b!} && \text{(by Eq (14) for the first term and Eq (15) for } \delta_i\text{-terms)}\\
&\leqslant \frac{20\sqrt{\theta}}{b!}, && \text{(as } \theta<1/100\text{)}
\end{aligned}
$$

concluding the proof. ∎

Conditioning Step: For the rest of the proof, we further condition on the choice of the set Good such that

$$|\textsc{Good}|\geqslant k/4, \tag{17}$$

and we know that by Eq (13), this event happens with high probability. We then condition on the entire choice of $\bar{\sigma}^{(i)}_{\ell_i,v_i}\in S_b$ for any $i\notin\textsc{Good}$. At this point, the only remaining variables which are not fixed are $\bar{\sigma}^{(i)}_{\ell_i,v_i}$ for $i\in\textsc{Good}$, which are still chosen independently by Lemma 5.8 (although we have conditioned on $i\in\textsc{Good}$, which influences their individual distribution).

5.3.3 Step III: Amplified ℓ_2-Distance for the Target Permutation

We now go over the last step of our proof by analyzing the distribution of the target permutation

$$\gamma^\star=\bar{\sigma}^{(1)}_{\ell_1,v_1}\circ\cdots\circ\sigma^{(k)}_{\ell_k,v_k};$$

in particular, we show that the distribution of $\gamma^\star$ is exponentially closer to the uniform distribution compared to the bounds of Lemma 5.10 as a result of concatenation (namely, that independent concatenation reduces the "bias").

To continue, we need some notation. Let $g:=|\textsc{Good}|$ and $i_1,\ldots,i_g$ be indices in Good sorted in increasing order. Define the following g permutations for $j\in[g]$ as concatenation of each $\bar{\sigma}^{(i_j)}_{\ell_{i_j},v_{i_j}}$ with all subsequent permutations which are not in Good until we hit the index i_{j+1}; formally,

$$\tilde{\sigma}_j:=\bar{\sigma}^{(i_j)}_{\ell_{i_j},v_{i_j}}\quad\circ\quad\bar{\sigma}^{(i_j+1)}_{\ell_{i_j+1},v_{i_j+1}}\quad\ldots\quad\circ\quad\bar{\sigma}^{(i_{j+1}-1)}_{\ell_{i_{j+1}-1},v_{i_{j+1}-1}};$$

Notice that the distribution of each $\tilde{\sigma}_j$ is the same as that of $\bar{\sigma}^{(i_j)}_{\ell_{i_j},v_{i_j}}\mid\bar{\Pi},i_j\in\textsc{Good}$ except for a fixed "shift" (by the fixed permutations indexed between $i_j,i_{j+1}\in\textsc{Good}$). Moreover, define

$$\tilde{\gamma}:=\tilde{\sigma}_1\circ\ldots\circ\tilde{\sigma}_g;$$

the distribution of $\tilde{\gamma}$ is now the same as that of the target permutation $\gamma^\star\mid\bar{\Pi},i_j\in\textsc{Good}$, again, except for a fixed shift (by the fixed permutations before $i_1\in\textsc{Good}$). Thus, for $j\in[g]$, we have,

$$
\begin{aligned}
\|\mu(\tilde{\boldsymbol{\sigma}}_j)-\mathcal{U}_{S_b}\|_2^2 &= \|\mu(\bar{\boldsymbol{\sigma}}^{(i_j)}_{\ell_{i_j},v_{i_j}}\mid\bar{\Pi},i_j\in\textsc{Good})-\mathcal{U}_{S_b}\|_2^2\leqslant\frac{20\sqrt{\theta}}{b!};\\
\|\mu(\tilde{\boldsymbol{\gamma}})-\mathcal{U}_{S_b}\|_2^2 &= \|\mu(\boldsymbol{\gamma}^\star\mid\bar{\Pi},\textsc{Good})-\mathcal{U}_{S_b}\|_2^2;\\
\mu(\tilde{\boldsymbol{\sigma}}_1,\ldots,\tilde{\boldsymbol{\sigma}}_g) &= \mu(\tilde{\boldsymbol{\sigma}}_1)\times\cdots\times\mu(\tilde{\boldsymbol{\sigma}}_g).
\end{aligned}
\tag{18}
$$

here, the first inequality of the first equation is by Lemma 5.10 and equality of the last equation is by Lemma 5.8.

We can now prove the following lemma on the distance of $\tilde{\gamma}$ from the uniform distribution, using a basic application of the Fourier analysis on permutations reviewed in Appendix B.

Lemma 5.11 ("Concatenation reduces ℓ_2-distances")**.**

$$\|\mu(\tilde{\gamma}) - \mathcal{U}_{S_b}\|_2^2 \leqslant \frac{(20\sqrt{\theta})^g}{b!}$$

Proof. Recall the definition of the Fourier transform and the set of irreducible representations for S_b from Appendix B. For the simplicity of exposition in this proof, we denote,

$$\nu_j := \mu(\tilde{\boldsymbol{\sigma}}_j) \text{ for every } j \in [g], \quad \text{and} \quad \nu := \mu(\tilde{\gamma}) = \nu_1 \circ \cdots \circ \nu_g.$$

For the distribution μ, by Plancherel's identity of Proposition B.5,

$$\begin{aligned}
\|\nu - \mathcal{U}_{S_b}\|_2^2 &= \sum_{\sigma \in S_b} \left(\nu(\sigma) - \frac{1}{b!}\right)^2 && \text{(by the definition of the } \ell_2\text{-norm)} \\
&= \frac{1}{b!} \cdot \sum_{\rho \in \textsc{RepBasis}} d_\rho \sum_{i,j \in [d_\rho]} \left(\widehat{\rho(\nu)} - \widehat{\rho(\mathcal{U}_{S_b})}\right)_{i,j}^2 && \text{(by Proposition B.5)} \\
&= \frac{1}{b!} \cdot \left(d_{\rho_0} \left(\widehat{\rho_0(\nu)} - \widehat{\rho_0(\mathcal{U}_{S_b})}\right)^2 + \sum_{\rho \neq \rho_0} d_\rho \sum_{i,j \in [d_\rho]} \left(\widehat{\rho(\nu)} - \widehat{\rho(\mathcal{U}_{S_b})}\right)_{i,j}^2 \right) \\
&\quad \text{(by splitting the outer sum over } \rho_0 \text{ and remaining representations in } \textsc{RepBasis}) \\
&= \frac{1}{b!} \cdot \left(\sum_{\rho \neq \rho_0} d_\rho \sum_{i,j \in [d_\rho]} \widehat{\rho(\nu)}_{i,j}^2 \right) \\
&\quad \text{(as } \widehat{\rho_0(\nu)} = \widehat{\rho(\mathcal{U}_{S_b})} = 1, \text{ and } \widehat{\rho(\mathcal{U}_{S_b})} = \mathbf{0} \in \mathbb{R}^{d_\rho \times d_\rho} \text{ for } \rho \neq \rho_0 \text{ by Fact B.3)} \\
&= \frac{1}{b!} \cdot \sum_{\rho \neq \rho_0} d_\rho \cdot \|\widehat{\rho(\nu)}\|_F^2, && (19)
\end{aligned}$$

where $\|\cdot\|_F$ denotes the Frobenius norm of the matrix $\widehat{\rho(\nu)} \in \mathbb{R}^{d_\rho \times d_\rho}$. Doing the same exact calculation for each ν_j for $j \in [g]$, we also get,

$$\sum_{\rho \neq \rho_0} d_\rho \cdot \|\widehat{\rho(\nu_j)}\|_F^2 = b! \cdot \|\nu_j - \mathcal{U}_{S_b}\|_2^2 \leqslant 20\sqrt{\theta}, \tag{20}$$

where the inequality is by Eq (18).

Finally, note that for every $\rho \in \textsc{RepBasis}$, by Fact B.4 (the convolution theorem),

$$\widehat{\rho(\nu)} = \prod_{j=1}^{g} \widehat{\rho(\nu_j)}.$$

Combining all these, we now have,

$$\|\nu - \mathcal{U}_{S_b}\|_2^2 = \frac{1}{b!} \cdot \sum_{\rho \neq \rho_0} d_\rho \cdot \|\widehat{\rho(\nu)}\|_F^2 \qquad \text{(by Eq (19))}$$

$$
\begin{aligned}
&= \frac{1}{b!} \cdot \sum_{\rho \neq \rho_0} d_\rho \cdot \| \prod_{j=1}^{g} \widehat{\rho(\nu_j)} \|_F^2 && \text{(by the application of Fact B.4 right above)} \\
&\leqslant \frac{1}{b!} \cdot \sum_{\rho \neq \rho_0} d_\rho \cdot \prod_{j=1}^{g} \|\widehat{\rho(\nu_j)}\|_F^2 && \text{(as Frobenius norm is sub-multiplicative)} \\
&\leqslant \frac{1}{b!} \cdot \sum_{\rho \neq \rho_0} \prod_{j=1}^{g} d_\rho \cdot \|\widehat{\rho(\nu_j)}\|_F^2 && \text{(as } d_\rho \geqslant 1\text{)} \\
&\leqslant \frac{1}{b!} \cdot \prod_{j=1}^{g} \sum_{\rho \neq \rho_t} d_\rho \cdot \|\widehat{\rho(\nu_j)}\|_F^2 && \text{(as } d_\rho, \|\widehat{\rho(\nu_j)}\|_F^2 \geqslant 0 \text{ for each } \rho \in \textsc{RepBasis} \text{ and } j \in [g]\text{)} \\
&\leqslant \frac{1}{b!} \cdot \prod_{j=1}^{g} (20\sqrt{\theta}) && \text{(by Eq (20))} \\
&= \frac{(20\sqrt{\theta})^g}{b!},
\end{aligned}
$$

concluding the proof. ∎

It is worth mentioning that the calculations we have in Lemma 5.11 are tight (see Appendix C for an example and more discussion).

Plugging back the original variables we had in Eq (18) inside Lemma 5.11, and using the bound of Eq (17) on the size of Good, we obtain that

$$
\|\mu(\boldsymbol{\gamma}^\star \mid \bar{\Pi}, \textsc{Good}) - \mathcal{U}_{S_b}\|_2^2 \leqslant \frac{1}{b!} \cdot (20\sqrt{\theta})^{k/4},
$$

which in turn, by the ℓ_1-ℓ_2 gap, gives us

$$
\begin{aligned}
\|\mu(\boldsymbol{\gamma}^\star \mid \bar{\Pi}, \textsc{Good}) - \mathcal{U}_{S_b}\|_{\text{tvd}} &\leqslant \|\mu(\boldsymbol{\gamma}^\star \mid \bar{\Pi}, \textsc{Good}) - \mathcal{U}_{S_b}\|_1 \\
&\qquad \text{(by the definition of total variation distance in Eq (30))} \\
&\leqslant \sqrt{b!} \cdot \|\mu(\boldsymbol{\gamma}^\star \mid \bar{\Pi}, \textsc{Good}) - \mathcal{U}_{S_b}\|_2 \\
&\qquad \text{(as } \|x\|_1 \leqslant \sqrt{m} \cdot \|x\|_2 \text{ for any } x \in \mathbb{R}^m\text{)} \\
&\leqslant (20\sqrt{\theta})^{k/8}. \qquad (21)
\end{aligned}
$$

We have thus finally bounded the TVD of the target permutation from the uniform distribution (under conditioning on several high probability events), and are now done with the proof.

5.4 Putting Everything Together: Proof of Theorem 2

We are now ready to conclude the proof of Theorem 2 by simply retracing back all the steps of the proof and accounting for the probability of several events we conditioned on. The rest of this proof is basically bookkeeping and some tedious calculations.

Recall that π is any deterministic protocol for **Multi-HPH** that uses s bits of communication in total. The proof consisted of the following:

- In Eq (10), we proved that,

$$
\Pr(\pi \text{ succeeds}) \leqslant \frac{1}{2} + \frac{r}{2} \cdot \max_{\substack{M_{<a}, \Sigma_{*, M_{<a}} \\ \mu = \mu_{M_{<a}, \Sigma_{*, M_{<a}}}}} \left[\mathop{\mathbb{E}}_{\bar{\Pi} \sim \mu} \mathop{\mathbb{E}}_{\substack{\ell_1, \ldots, \ell_k \sim \mu \\ v_1, \ldots, v_k}} \|\mu(\bar{\boldsymbol{\sigma}}^{(1)}_{\ell_1, v_1} \circ \cdots \circ \bar{\boldsymbol{\sigma}}^{(k)}_{\ell_k, v_k} \mid \Pi) - \mathcal{U}_{S_b}\|_{\text{tvd}} \right].
$$

We then fixed any choice of $M_{<a}, \Sigma_{*,M_{<a}}$ and $\mu = \mu_{M_{<a},\Sigma_{*,M_{<a}}}$ for the rest of the proof.

- In Eq (12), we fixed any choice of $(\bar{\Pi}, L, e)$ that guarantees
$$\textsc{Info} := \textsc{Info}(\bar{\Pi}, L, e) \leqslant k/2.$$
By Lemma 5.5, the probability of $(\bar{\Pi}, L, e) \sim \mu$ not satisfying this is at most $2 \cdot (64\theta)^{k/4}$.
- In Eq (17), we fixed any choice of $\textsc{Good}$ that guarantees
$$|\textsc{Good}| \geqslant k/4.$$
By Eq (13), the probability that $\textsc{Good}$ does not satisfy this guarantee is at most $(128\theta)^{k/8}$.
- In Eq (21), we proved that for this choice of $(\bar{\Pi}, L, e)$ and conditioned on the choice of $\textsc{Good}$,
$$\|\mu(\bar{\boldsymbol{\sigma}}^{(1)}_{\ell_1,v_1} \circ \cdots \circ \bar{\boldsymbol{\sigma}}^{(k)}_{\ell_k,v_k} \mid \bar{\Pi}, \textsc{Good}) - \mathcal{U}_{S_b}\|_{\text{tvd}} \leqslant (20\sqrt{\theta})^{k/8} \leqslant 2 \cdot \theta^{k/16}.$$

Let us now retrace back.

- For any $M_{<a}, \Sigma_{*,M_{<a}}$, $\mu = \mu_{M_{<a},\Sigma_{*,M_{<a}}}$, and $(\bar{\Pi}, L, e)$ satisfying $\textsc{Info}(\bar{\Pi}, L, e) \leqslant k/2$, we have,
$$\begin{aligned}
&\|\mu(\bar{\boldsymbol{\sigma}}^{(1)}_{\ell_1,v_1} \circ \cdots \circ \bar{\boldsymbol{\sigma}}^{(k)}_{\ell_k,v_k} \mid \bar{\Pi}) - \mathcal{U}_{S_b}\|_{\text{tvd}} \\
&\quad\leqslant \Pr\left(|\textsc{Good}| < k/4\right) + \mathop{\mathbb{E}}_{\substack{\textsc{Good}\sim\mu| \\ |\textsc{Good}|\geqslant k/4}} \|\mu(\bar{\boldsymbol{\sigma}}^{(1)}_{\ell_1,v_1} \circ \cdots \circ \bar{\boldsymbol{\sigma}}^{(k)}_{\ell_k,v_k} \mid \bar{\Pi}, \textsc{Good}) - \mathcal{U}_{S_b}\|_{\text{tvd}} \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{(by Fact A.10 and since } \|\cdot\|_{\text{tvd}} \leqslant 1) \\
&\quad\leqslant (128\theta)^{k/8} + 2 \cdot \theta^{k/16}
\end{aligned}$$
- This in turn implies that for any $M_{<a}, \Sigma_{*,M_{<a}}$, $\mu = \mu_{M_{<a},\Sigma_{*,M_{<a}}}$,
$$\begin{aligned}
&\mathop{\mathbb{E}}_{\bar{\Pi},L,e\sim\mu} \|\mu(\bar{\boldsymbol{\sigma}}^{(1)}_{\ell_1,v_1} \circ \cdots \circ \bar{\boldsymbol{\sigma}}^{(k)}_{\ell_k,v_k} \mid \Pi) - \mathcal{U}_{S_b}\|_{\text{tvd}} \\
&\quad\leqslant \Pr_{\mu}\left(\left|\textsc{Info}(\bar{\Pi}, L, e)\right| > k/2\right) + \mathop{\mathbb{E}}_{\substack{(\bar{\Pi},L,e)\sim\mu| \\ \left|\inf(\bar{\Pi},L,e\right|\leqslant k/2}} \|\mu(\bar{\boldsymbol{\sigma}}^{(1)}_{\ell_1,v_1} \circ \cdots \circ \bar{\boldsymbol{\sigma}}^{(k)}_{\ell_k,v_k} \mid \bar{\Pi}) - \mathcal{U}_{S_b}\|_{\text{tvd}} \\
&\quad\leqslant 2 \cdot (64\theta)^{k/4} + (128\theta)^{k/8} + 2 \cdot \theta^{k/16}.
\end{aligned}$$
- Finally, we can plug in this bound in Eq (10) and have,
$$\begin{aligned}
\Pr\left(\pi \text{ succeeds}\right) &\leqslant \frac{1}{2} + \frac{r}{2} \cdot \max_{\substack{M_{<a},\Sigma_{*,M_{<a}} \\ \mu=\mu_{M_{<a},\Sigma_{*,M_{<a}}}}} \mathop{\mathbb{E}}_{\bar{\Pi},L,e\sim\mu} \|\mu(\bar{\boldsymbol{\sigma}}^{(1)}_{\ell_1,v_1} \circ \cdots \circ \bar{\boldsymbol{\sigma}}^{(k)}_{\ell_k,v_k} \mid \Pi) - \mathcal{U}_{S_b}\|_{\text{tvd}} \\
&\leqslant \frac{1}{2} + r \cdot O(\theta)^{k/16} \\
&\leqslant \frac{1}{2} + r \cdot O\left(\frac{8s + 8k \cdot \log(t \cdot r)}{t \cdot r}\right)^{k/32} \\
&\qquad \text{(by the definition of } \theta \text{ in Definition 5.4 and since } r_a \geqslant r/2 \text{ by Claim 5.3-Item } (ii)) \\
&\leqslant \frac{1}{2} + r \cdot O\left(\frac{s}{r \cdot t}\right)^{k/32}. \quad \text{(by the lower bound on the size of } s \text{ in Theorem 2)}
\end{aligned}$$

This concludes the proof for all deterministic protocols π on the distribution induced by **Multi-HPH**. The lower bound directly extends to randomized protocols by the easy direction of Yao's minimax principle (namely, an averaging argument over the randomness of the protocol on the input distribution). This concludes the proof of Theorem 2. ∎

6 One Pass Permutation Hiding

In this section, we will construct permutation hiding graphs for 1-pass streaming algorithms. We repeat the key definitions here (Definition 2.1 and Definition 2.2) for the convenience of the reader.

Definition (Permutation Graph)**.** For any integer $m \geqslant 1$, a layered graph $G = (V, E)$ is said to be a **permutation graph** for $\sigma \in S_m$ if $|\textsc{First}(G)|, |\textsc{Last}(G)| \geqslant m$ and there is a path from $i \in \textsc{First}_{[m]}(G)$ to $j \in \textsc{Last}_{[m]}(G)$ if and only if $\sigma(i) = j$ for each $i, j \in [m]$.

Definition (Permutation Hiding Graphs; c.f. [CKP+21])**.** For integers $m, n, p, s \geqslant 1$ and real $\delta \in (0, 1)$, we define a **permutation hiding generator** $\mathcal{G} = \mathcal{G}(m, n, p, s, \delta)$ as any family of distributions $\mathcal{G} : S_m \to \mathcal{D}_m$ on permutation graphs satisfying the following two properties:

(*i*) For any $\sigma \in S_m$, any permutation graph G in the support of $\mathcal{G}(\sigma)$ has n vertices.

(*ii*) For any $\sigma_1, \sigma_2 \in S_m$, the distribution of graphs $\mathcal{G}(\sigma_1)$ and $\mathcal{G}(\sigma_2)$ are δ-indistinguishable for any p-pass s-space streaming algorithm.

We need some further definitions also. Given two layered graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$, we define their **concatenation**, $G_1 \circ G_2 = (V_1 \cup V_2, E)$ as follows. Let $\ell_1 = |\textsc{Last}(G_2)|$ and $\ell_2 = |\textsc{First}(G_1)|$. The edge set E is made up of $E_1 \cup E_2$ and the identity perfect matching from the set $\textsc{Last}(G_2)$ to the set $\textsc{First}_{[\ell_1]}(G_1)$ if $\ell_1 \leqslant \ell_2$ and identity perfect matching from $\textsc{Last}_{[\ell_2]}(G_2)$ to $\textsc{First}(G_1)$ if $\ell_1 > \ell_2$.

Concatenating permutation graphs gives us a graph for the concatenated permutation.

Claim 6.1. *Given G_1, G_2 which are permutation graphs for $\sigma_1, \sigma_2 \in S_m$ respectively, $G_1 \circ G_2$ is a permutation graph for the permutation $\sigma_1 \circ \sigma_2 \in S_m$.*

Proof. For any vertex $i \in \textsc{First}_{[m]}(G_2)$, there is a path to $\sigma_2(i) \in \textsc{Last}_{[m]}(G_2)$ by Definition 2.1. There is an edge from $\sigma_2(i) \in \textsc{Last}_{[m]}(G_2)$ to $\sigma_2(i) \in \textsc{First}_{[m]}(G_1)$ by the addition of the identity perfect matching. Again by Definition 2.1, there is a path from $\sigma_2(i) \in \textsc{First}_{[m]}(G_1)$ to $\sigma_1(\sigma_2(i)) \in \textsc{Last}_{[m]}(G_1)$ for each $i \in [m]$.

Thus, the path from $i \in \textsc{First}_{[m]}(G_1 \circ G_2)$ to $\sigma_1(\sigma_2(i)) \in \textsc{Last}_{[m]}(G_1 \circ G_2)$ exists for each $i \in [m]$, and no other path from any vertex in $\textsc{First}_{[m]}(G_1 \circ G_2)$ to $\textsc{Last}_{[m]}(G_1 \circ G_2)$ exists. ∎

The main lemma of this section follows.

Lemma 6.2. *There exists a permutation hiding generator $\mathcal{G} : S_m \to \mathcal{D}_m$ for 1-pass streaming algorithms using space $s = o(m^{1+\beta/2})$ such that*

$$n \leqslant 2 \cdot c_{\text{SORT}} \cdot 10^6 \cdot (m/\alpha\beta^2)$$
$$\delta \leqslant 100 \cdot c_{\text{SORT}} \cdot (1/\beta) \cdot m^{-5} \cdot (\alpha^\beta \cdot \beta^{-1})^{50/\beta},$$

where α, β are from Eq (6).

In the first subsection, we define the constructs required to describe our permutation hiding graphs and in the next subsection we prove that our graphs can hide simple permutations. In the final subsection, we extend our graphs to hide any general permutation from one pass semi-streaming algorithms.

6.1 Building Blocks for Permutation Hiding

In this section, we will define the constructs needed to describe our permutation hiding graphs. The main parts are as follows.

- **Group layered graphs, encoded RS graphs, and other helper structures**: these are simple permutation graphs needed for our main construction;
- **Blocks**: These are permutation graphs that "encode" a single permutation inside it;
- **Multi-blocks**: These are permutation graphs that hide the concatenation of several permutations (via concatenation of several blocks).

Group layered graphs, encoded RS graphs, and other helper structures

The most basic permutation graph consisting of only two layers is defined first.

Definition 6.3 (Basic Permutation Graph)**.** Given a permutation $\sigma \in S_m$, we define a graph $G = (L \cup R, E)$ as a **basic permutation graph** of σ, denoted by $\textsc{Basic}(\sigma)$ if $|L| = |R| = m$ and $E = \{(i, \sigma(i)) \mid i \in [m]\}$.

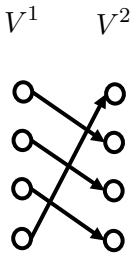


Figure 7: An illustration of $\textsc{Basic}(\sigma)$ from Definition 6.3 with $\sigma = (2, 3, 4, 1)$.

We need to define some more structure on layered graphs.

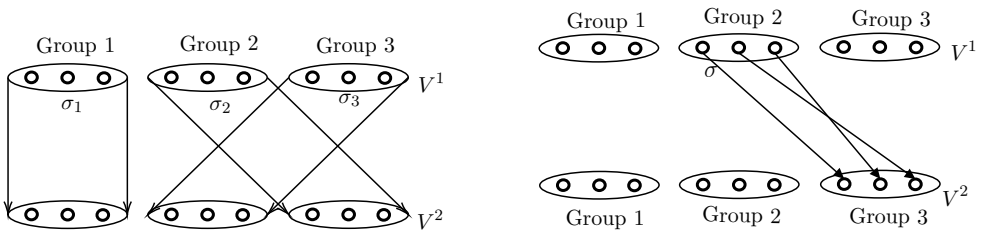
Definition 6.4 (Group-Layered Graph)**.** For integers $w, d, b \geqslant 1$, we define a **group-layered graph** as any directed acyclic graph $G = (V, E)$ satisfying the following:

(*i*) Vertices of G are partitioned into d equal-size **layers** $V^1, \ldots, V^d$, each of size $w \cdot b$. We identify each layer with the pairs in $[w] \times [b]$.

(*ii*) For any layer $i \in [d]$ and $a \in [w]$, we define the **group** $V^{i,a} := \{a\} \times [b]$.

(*iii*) Edges of G can be defined via some tuples $(i, a_1, a_2, \sigma) \in [d] \times [w] \times [w] \times S_b$ as follows: We connect $(a_1, j) \in V^i$ to $(a_2, \sigma(j)) \in V^{i+1}$ for all $j \in [b]$.

We refer to w as the **width** of the layered graph, to d as its **depth**, and b as its **group size**. We use $\mathcal{L}_{w,d,b}$ to denote the set of all layered graphs with width, depth, and group size, w, d, and b, respectively. See Figure 8 for an illustration.



(a) A (partially drawn) group layered graph in $\mathcal{L}_{3,2,3}$ with 2 layers and each layer identified by $[3] \times [3]$ with edges identified by $(1, 1, 1, \sigma_1), (1, 2, 3, \sigma_2), (1, 3, 2, \sigma_3)$.

(b) An explicit edge in $\mathcal{L}_{3,2,3}$ identified by $(1, 2, 3, \sigma_2)$ with $\sigma_2 = (1, 3, 2)$.

Figure 8: An illustration of a group layered graph from Definition 6.4.

While performing concatenation on a group layered graph $G \in \mathcal{L}_{w,d,b}$ with any other graph G', we treat the vertices in $\textsc{First}(G)$, indexed by $[w] \times [b]$, as being indexed by the set $[w \cdot b]$ by directly mapping any vertex (i, j) to $(i - 1) \cdot b + j$ for $i \in [w], j \in [b]$.

Group-layered graphs allow us to capture two main ideas: (i) Reachability between multiple groups; and (ii) Permuting within each group. We formalize this in the following.

Consider any tuple $(i, a_1, a_2, \sigma_{\text{ID}}) \in [d] \times [w] \times [w] \times S_b$. The edges associated with this tuple simply connect group V^{i,a_1} to V^{i+1,a_2} with the identity permutation. Such edges allow us to *add paths between the groups* of a group-layered graph. We formalize this concept next.

Definition 6.5 (Group Permuting Graph)**.** Given a permutation $\sigma \in S_w$ and any integer $b \geqslant 1$, we define the **group permuting graph**, denoted by $\textsc{Permute}(\sigma, b) = (V, E)$ as the group layered graph from $\mathcal{L}_{w,2,b}$ with the edges $(1, i, \sigma(i), \sigma_{\text{ID}}(b))$ for each $i \in [w]$ (see Figure 9).

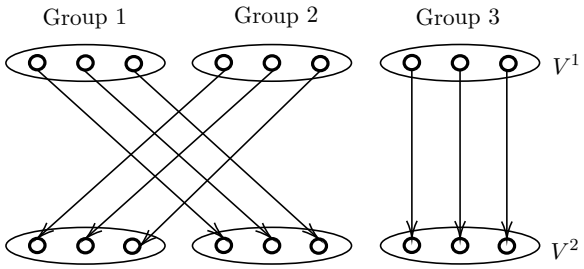


Figure 9: A group permuting graph from Definition 6.5 with $\sigma = (2, 1, 3)$ and $b = 3$.

The second main idea in group-layered graphs is to permute within the groups. Here, for any edge tuple (i, a_1, a_2, σ), if a_1, a_2 are fixed, $\sigma \in S_b$ allows us to *permute within the groups*. The following definition is an instance of permuting within the groups.

Definition 6.6 (Encoded RS graph)**.** Given an (r, t)-RS-graph $G_{\text{rs}} = (L_{\text{rs}} \cup R_{\text{rs}}, E_{\text{rs}})$, and a permutation matrix $\Sigma \in (S_b)^{t \times r}$, the **encoded RS graph**, denoted by $\textsc{Encoded-RS}(G_{\text{rs}}, \Sigma) = (V, E)$ is a group layered graph from $\mathcal{L}_{n^{\text{rs}},2,b}$ with the edges $(1, \textsc{left}(j), \textsc{right}(j), \sigma_{i,j})$ for edge $j \in M_i^{\text{rs}}$, for each $i \in [t], j \in [r]$ (see Figure 10).

Observe that if the RS-graph G_{rs} is fixed, the tuple $(\textsc{left}(j), \textsc{right}(j))$ is fixed for edge $j \in M_i^{\text{rs}}$ for each $i \in [t], j \in [r]$. Based on these fixed tuples, we are permuting within the groups based on

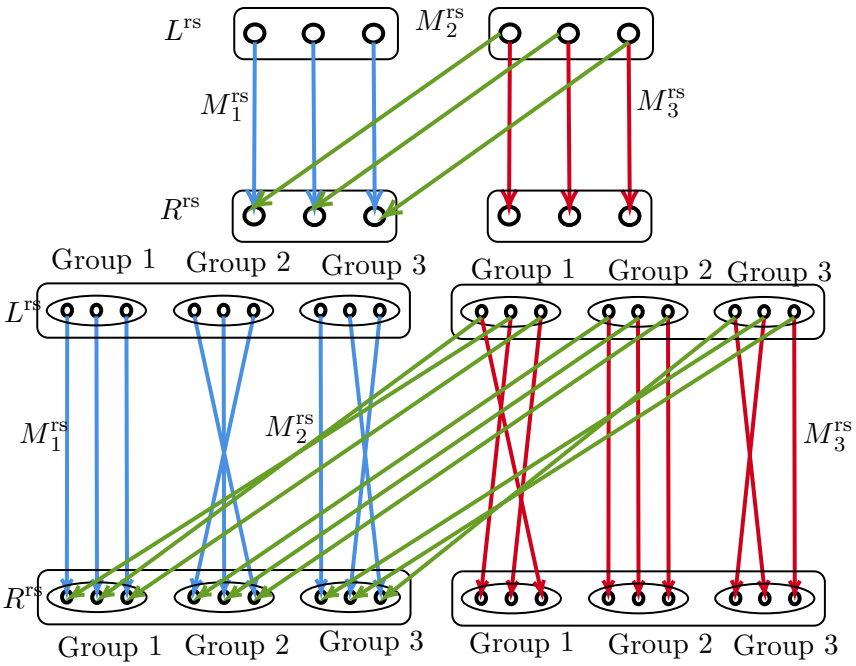


Figure 10: An illustration of an encoded (r, t) RS-graph with $r = 3, t = 3$ from Definition 6.6 with $\Sigma \in (S_b)^{3\times 3}$ and $\sigma_{1,1} = (1, 2, 3), \sigma_{1,2} = (3, 2, 1), \sigma_{1,3} = (1, 3, 2)$ as blue edges; $\sigma_{2,1} = (2, 1, 3), \sigma_{2,2} = (1, 2, 3), \sigma_{2,3} = (3, 1, 2)$ as green edges; $\sigma_{3,1} = (3, 1, 2), \sigma_{3,2} = (1, 2, 3), \sigma_{3,3} = (2, 1, 3)$ as red edges.

matrix Σ. We combine the two main ideas when we construct blocks. Before we proceed, we need a few more definitions. The first is that of a permutation that agrees with a given matching.

Definition 6.7 (Match-Aligned Permutation)**.** Given a matching M on $G = (L \cup R, E)$ with $|L| = |R| = m$, $\sigma \in S_m$ is said to be a **match-aligned permutation**, denoted by $\textsc{Match-Perm}(M)$ if it is the lexicographically first permutation with $\sigma(u) = v$ for all $(u, v) \in M$.

In other words, we view any matching as a partial permutation where the edges in the matching fix certain assignments of the permutation, and Definition 6.7 extends this to a complete permutation. We also need permutations that pick certain edges from a given RS-graph.

Definition 6.8 (Edge Picking Permutations)**.** Given an (r, t)-RS-graph $G_{\text{rs}} = (L_{\text{rs}} \cup R_{\text{rs}}, E_{\text{rs}})$, an index $\ell \in [t]$ for an induced matching M_ℓ^{rs} in G_{rs} and $(r/2)$ edges $E^* = (e_1, e_2, \ldots, e_{r/2})$ from matching M_ℓ^{rs}, define matchings $M_{\text{L}}, M_{\text{R}}$ on vertices $L \cup R$ with $|L| = |R| = n^{\text{rs}}$ obeying,

$$M_{\text{L}} = \{(i, \textsc{left}(e_i)) \mid i \in [r/2]\} \qquad \text{and} \qquad M_{\text{R}} = \{(\textsc{right}(e_i), i) \mid i \in [r/2]\}.$$

Then, **edge picking permutations** (see Figure 11), denoted by $\textsc{Edge-Pick}(G_{\text{rs}}, \ell, E^*)$ is an ordered pair $(\sigma_{\text{L}}, \sigma_{\text{R}}) \in S_{n^{\text{rs}}} \times S_{n^{\text{rs}}}$, where

$$\sigma_{\text{L}} = \textsc{Match-Perm}(M_{\text{L}}) \qquad \text{and} \qquad \sigma_{\text{R}} = \textsc{Match-Perm}(M_{\text{R}}).$$

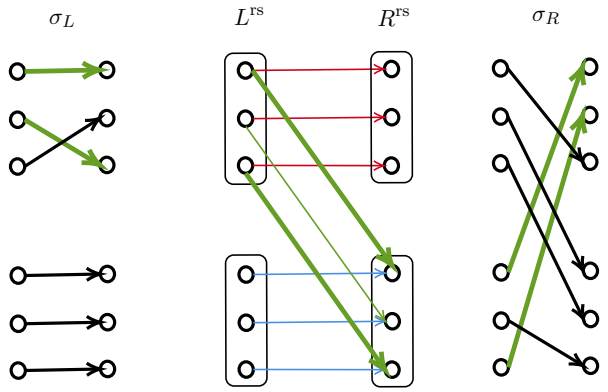


Figure 11: An illustration of edge picking permutations from Definition 6.8. The RS-graph from Figure 10 is picked with $\ell = 2$ and (bold) edges $e_1, e_2 \in M_2^{\text{rs}}$. The permutation σ_{L} and σ_{R} are $(1,3,2,4,5,6)$ and $(3,4,5,1,6,2)$, respectively.

Blocks

A key gadget in our construction, called a block is built using the preceding definitions. It combines the two main ideas from group layered graphs.

Definition 6.9 (Block)**.** For any $r, t, b \geqslant 1$, given

- An (r,t) RS-graph $G_{\text{rs}} = (L_{\text{rs}} \cup R_{\text{rs}}, E_{\text{rs}})$ with $L_{\text{rs}} = R_{\text{rs}} = [n^{\text{rs}}]$ and t induced matchings $M_1^{\text{rs}}, \dots M_t^{\text{rs}}$,
- A permutation matrix $\Sigma \in (S_b)^{t \times r}$,
- An index $\ell \in [t]$ and $(r/2)$ edge tuple $E^* = (e_1, e_2, \dots, e_{r/2})$ with each edge belonging to M_ℓ^{rs},

let $(\sigma_{\text{L}}, \sigma_{\text{R}}) \in S_{n^{\text{rs}}} \times S_{n^{\text{rs}}}$ be Edge-Pick$(G_{\text{rs}}, \ell, E^*)$. We define a **block** (see Figure 12), denoted by Block$(G_{\text{rs}}, \Sigma, \ell, E^*)$ as,

$$\text{Permute}(\sigma_{\text{R}}, b) \circ \text{Encoded-RS}(G_{\text{rs}}, \Sigma) \circ \text{Permute}(\sigma_{\text{L}}, b).$$

Let us prove some useful properties of blocks.

Claim 6.10. *The graph* Block$(G_{\text{rs}}, \Sigma, \ell, E^*)$ *is a layered graph in* $\mathcal{L}_{n^{\text{rs}},6,b}$.

Proof. The graph Block$(G_{\text{rs}}, \Sigma, \ell, E^*)$ is made of concatenating three group layered graphs, Permute(σ_{R}, b), Encoded-RS(G_{rs}, Σ), and Permute(σ_{L}, b). As both $\sigma_{\text{L}}, \sigma_{\text{R}}$ are permutations from $S_{n^{\text{rs}}}$, Permute(σ_{L}, b) and Permute(σ_{R}, b) both belong to $\mathcal{L}_{n^{\text{rs}},2,b}$. By Definition 6.6, as Encoded-RS$(G_{\text{rs}}, \Sigma) \in \mathcal{L}_{n^{\text{rs}},2,b}$, the concatenation of all three graphs belongs to $\mathcal{L}_{n^{\text{rs}},6,b}$. ∎

Next, let us see how the edges in a block between various layers are determined by the inputs.

Claim 6.11. *In any graph* $G =$ Block$(G_{\text{rs}}, \Sigma, \ell, E^*)$ *with layers* $V^1, V^2, \dots, V^6$ *(see Claim 6.10),*

(*i*) *The edges between* V^3 *to* V^4 *are determined only by* G_{rs} *and* Σ.

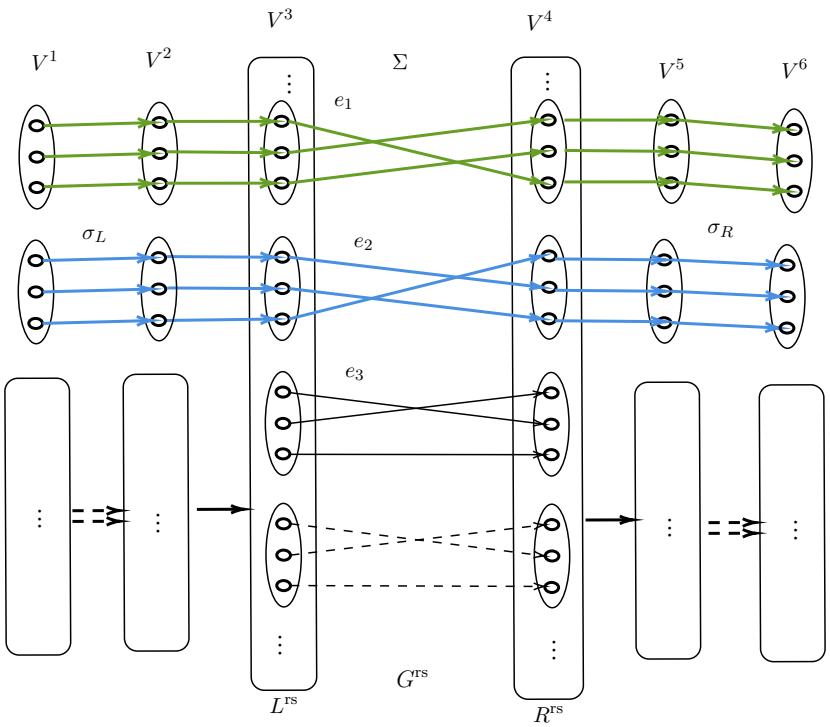


Figure 12: An illustration of block from Definition 6.9 with $r = 3, b = 3$ showing the choosing of one induced matching $M^{\text{rs}}_{j^*}$ with edges e_1 (green) and e_2 (blue). The dashed light edge in G_{rs} represents an edge in another induced matching. The edges from V^1 to V^2 are from $\textsc{Permute}(\sigma_{\text{L}}, b)$, the edges from V^5 to V^6 are from $\textsc{Permute}(\sigma_{\text{R}}, b)$. The edges from V^3 to V^4 are from $\textsc{Encoded-RS}(G_{\text{rs}}, \Sigma)$.

(*ii*) *The edges from layers V^1 to V^2 and from V^5 to V^6 are fixed by G_{rs}, ℓ and E^*.*

(*iii*) *The other edges from V^2 to V^3 and from V^4 to V^5 are fixed perfect matchings.*

Proof. The edges from V^3 to V^4 are from $\textsc{Encoded-RS}(G_{\text{rs}}, \Sigma)$, and hence depend only on G_{rs} and Σ. The edges between V^1 to V^2 and from V^5 to V^6 are based on permutations σ_{L} and σ_{R} from Definition 6.8, and depend only on G_{rs}, ℓ and E^*. The other edges are added when concatenating group-layered graphs, and as the layers have same size $[n^{\text{rs}} \cdot b]$, they are perfect matchings. ∎

Let Lex be an equipartition of the set $[m]$ into m/b groups of size b each, partitioning the elements lexicographically throughout this section. That is, for each $i \in [m/b]$, the set $\mathsf{Lex}_i = \{(i-1)b + a \mid a \in [b]\}$ is a group in partition Lex. Recall from Definition 4.2 that a simple permutation $\rho \in S_m$ on partition Lex is such that for all $a \in \mathsf{Lex}_i$, $\rho(a) \in \mathsf{Lex}_i$. Given Σ, ℓ and $E^* = (e_1, e_2, \ldots, e_{r/2})$ with $r/2 = m/b$ edges from M^{rs}_ℓ as the inputs from Definition 6.9 , let ρ be the following simple permutation on partition Lex:

$$\forall i \in [m/b], a \in [b], \rho((i-1)b + a) = (i-1)b + \sigma_{\ell,e_i}(a). \tag{22}$$

Claim 6.12. $\textsc{Block}(G_{\text{rs}}, \Sigma, \ell, E^*)$ *is a permutation graph for ρ defined in Eq (22).*

Proof. We know that each vertex $(i, a) \in V^1$ for $i \in [m/b], a \in [b]$, also indexed by $(i-1)b+a \in [m]$ is connected to $(\sigma_{\text{L}}(i), a) \in V^2$ by the edge added from $\textsc{Permute}(\sigma_{\text{L}})$. By Definition 6.8, we know that $\sigma_{\text{L}}(i) = \textsc{left}(e_i)$ for $i \in [r/2]$ and edge $e_i \in M^{\text{rs}}_\ell$. The vertex $(\textsc{left}(e_i), a) \in V^3$ is

connected to $(\text{RIGHT}(e_i), \sigma_{\ell,e_i}(a)) \in V^4$ by the edges from $\text{ENCODED-RS}(G_{\text{rs}}, \Sigma)$. Lastly, the vertex $(\text{RIGHT}(e_i), \sigma_{\ell,e_i}(a)) \in V^5$ is connected to $(i, \sigma_{\ell,e_i}(a)) \in V^6$ based on permutation σ_{R}. The proof is complete when we also look at the identity perfect matchings connecting layers V^2 to V^3 and layers V^4 to V^5. ∎

Multi-blocks

The next step is to combine multiple blocks to get a larger graph and to simulate the concatenation of multiple simple permutations.

Definition 6.13 (Multi-Block)**.** For any $r, t, b, k \geqslant 1$, given

- An (r, t) RS-graph $G_{\text{rs}} = (L_{\text{rs}} \cup R_{\text{rs}}, E_{\text{rs}})$ with $L_{\text{rs}} = R_{\text{rs}} = [n^{\text{rs}}]$ and t induced matchings $M_1^{\text{rs}}, \dots M_t^{\text{rs}}$,
- A collection of k permutation matrices $\Sigma = (\Sigma^{(1)}, \Sigma^{(2)}, \dots, \Sigma^{(k)}) \in \left((S_b)^{t \times r}\right)^k$,
- A tuple $L = (\ell_1, \ell_2, \dots, \ell_k) \in [t]^k$ and a hypermatching M on the k-layered hypergraph $[r]^k$ of size $r/2$,

let E_i^* be $(M_{1,i}, M_{2,i}, \dots, M_{r/2,i})$ be $r/2$ edges in $M_{\ell_i}^{\text{rs}}$ for $i \in [k]$. We define a **multi-block** (see Figure 13), denoted by $\text{MULTI-BLOCK}(G_{\text{rs}}, \Sigma, L, M)$ as,

$$\text{BLOCK}(G_{\text{rs}}, \Sigma^{(1)}, \ell_1, E_1^*) \circ \text{BLOCK}(G_{\text{rs}}, \Sigma^{(2)}, \ell_2, E_2^*) \circ \dots \circ \text{BLOCK}(G_{\text{rs}}, \Sigma^{(k)}, \ell_k, E_k^*).$$

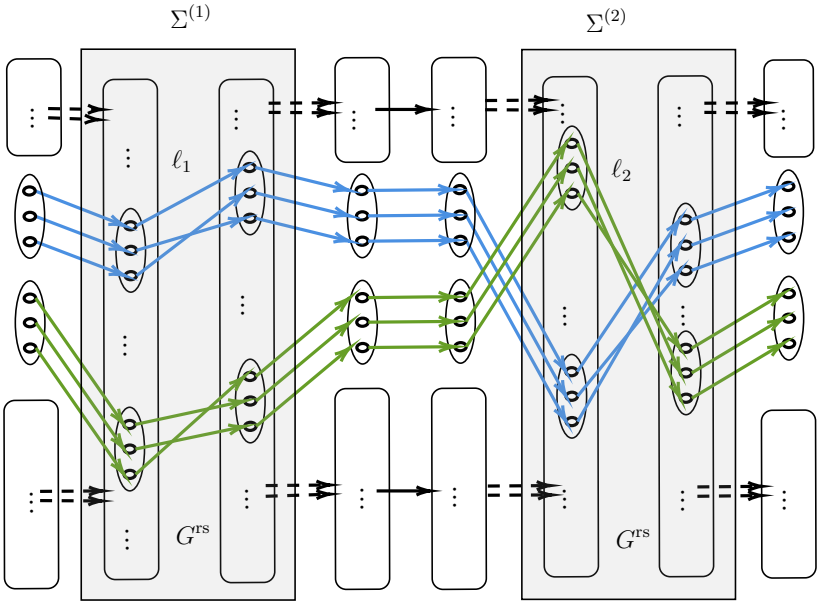


Figure 13: An illustration of multi-block from Definition 6.13 with 2 blocks. The blue path corresponds to one specific edge in each induced matching, and the green corresponds to the other path.

We can extend the properties we proved about blocks to multi-blocks.

Claim 6.14. *The graph* $\text{MULTI-BLOCK}(G_{\text{rs}}, \Sigma, L, M)$ *belongs to* $\mathcal{L}_{n^{\text{rs}}, 6k, b}$.

Proof. Follows directly from Claim 6.10, and Definition 6.13, as we concatenate k blocks. ∎

Claim 6.15. *In any graph* $G = \text{Multi-Block}(G_{\text{rs}}, \Sigma, L, M)$,

(*i*) *For any* $i \in [k]$, *the edges between layer* $V^{(i-1)6+3}$ *to* $V^{(i-1)6+4}$ *are determined only by* $G_{\text{rs}}, \Sigma^{(i)}$.

(*ii*) *The edges between* $V^{(i-1)6+1}$ *to* $V^{(i-1)6+2}$ *and between* $V^{(i-1)6+5}$ *to* V^{6i} *for any* $i \in [k]$ *are determined by* G_{rs}, L, M.

(*iii*) *All the other edges are fixed identity perfect matchings.*

Proof. For any $i \in [k]$, the edges between $V^{(i-1)6+3}$ to $V^{(i-1)6+4}$ come from edges between layer V^3 to V^4 of $\text{Block}(G_{\text{rs}}, \Sigma^{(i)}, \ell_i, E_i^*)$, and by Claim 6.11, are determined only by $G_{\text{rs}}, \Sigma^{(i)}$. The edges between $V^{(i-1)6+1}$ to $V^{(i-1)6+2}$ and from $V^{(i-1)6+5}$ to V^{6i} are determined by $G_{\text{rs}}, \ell_i, E_i^*$ for $i \in [k]$, again by Claim 6.11, and thus depend only on G_{rs}, L, M. Finally, in the concatenations, we add identity-perfect matchings between the layers. ∎

For $i \in [k]$, $\Sigma^{(i)}, \ell_i$ and $E_i^* = (e_1, e_2, \ldots, e_{r/2})$ as in Definition 6.13 , let $\rho_i \in S_m$ be the following simple permutation on partition Lex:

$$\forall j \in [m/b], a \in [b], \rho_i((j-1)b + a) = (j-1)b + \sigma_{\ell_i, e_j}(a). \tag{23}$$

Let $\gamma^\star \in S_m$ be another simple permutation on Lex defined as,

$$\gamma^\star = \rho_1 \circ \rho_2 \circ \ldots \circ \rho_k. \tag{24}$$

Lemma 6.16. $\text{Multi-Block}(G_{\text{rs}}, \Sigma, L, M)$ *is a permutation graph of* $\gamma^\star$.

Proof. By Claim 6.12, we know that $\text{Block}(G_{\text{rs}}, \Sigma^{(i)}, \ell_i, E_i^*)$ is a permutation graph for $\rho_i \in S_m$ for each $i \in [k]$. By Claim 6.1, $\text{Multi-Block}(G_{\text{rs}}, \Sigma, L, M)$ is a permutation graph for $\gamma^\star$. ∎

This concludes our subsection for describing the building blocks and constructs needed for our permutation hiding generators.

6.2 Simple Permutation Hiding in One Pass

In this section, we will construct permutation hiding generators for simple permutations. Let us start by defining them.

Definition 6.17. For any integers $n, p, s \geqslant 1$, partition $\mathcal{P}$ of $[m]$ into $m/b \geqslant 1$ blocks of size $b \geqslant 1$, and error parameter $\delta \in [0, 1]$, a simple permutation hiding generator $\mathcal{G}^{\text{SIM}}(n, p, s, \mathcal{P}, \delta)$ is defined as a function $\mathcal{G}^{\text{SIM}}$ from the set of all simple permutations under partition $\mathcal{P}$ to $\mathcal{D}_m$ such that,

(*i*) For any $\rho \in S_m$ which is simple under $\mathcal{P}$, any $G \sim \mathcal{G}^{\text{SIM}}(\rho)$ is a permutation graph for ρ with n vertices.

(*ii*) For any $\rho_1, \rho_2 \in S_m$, both simple under $\mathcal{P}$, the two distributions $\mathcal{G}^{\text{SIM}}(\rho_1)$ and $\mathcal{G}^{\text{SIM}}(\rho_2)$ are δ-indistinguishable for p-pass s-space streaming algorithms.

In this subsection, we will prove that such generators can be constructed for the partition Lex.

Lemma 6.18. *For any $b \geqslant 2$, there is a simple permutation hiding generator $\mathcal{G}^{\text{SIM}} : S_m^{\text{SIM}} \to \mathcal{D}_m$ with respect to partition* Lex *for 1-pass streaming algorithms using space $s = o(\left(\frac{m}{b}\right)^{1+2\beta/3})$ with parameters $n = (2 \cdot 10^4 \cdot m/\alpha\beta)$ and $\delta = (2m/b)^{-6} \cdot (\alpha^\beta \cdot \beta^{-1})^{50/\beta}$.*

We will construct these generators using multi-blocks, and the inputs to the multi-blocks are from instances of **Multi-HPH**. By Observation 4.4, it is sufficient to hide multiple smaller permutations from S_b (m/b of them, to be exact) to hide any simple permutation. We will be talking about one instance of Problem 1 in the construction, and we will use $\Sigma \in \left((S_b)^{t\times r}\right)^k, L \in [t]^k$, hypermatching $M \subset [r]^k$ and $\Gamma^*, \Gamma_{\text{Yes}}, \Gamma_{\text{No}}, \Gamma \in (S_b)^{r/2}$ to denote the variables in the instance. See Figure 14 for an illustration of this construction.

Construction of $\mathcal{G}^{\text{SIM}}(n, p, s, \text{Lex}, \delta) : S_m^{\text{SIM}} \to \mathcal{D}_m$ on input $\rho \in S^{\text{SIM}}$:

(*i*) Fix an (r, t)-RS graph G_{rs} with $r = 2m/b = n^{\text{rs}}/\alpha$ and $t = (n^{\text{rs}})^\beta$.

(*ii*) Instantiate k players and the referee in **Multi-HPH**$_{r,t,b,k}$ (Problem 1) with parameter $k = 1600/\beta$ such that the solution to the instance, $\Gamma^* \circ \Gamma$ is $\text{VEC}(\rho)$.

(*iii*) Output $\text{MULTI-BLOCK}(G_{\text{rs}}, \Sigma, L, M) \circ \text{BASIC}(\text{JOIN}(\Gamma))$.

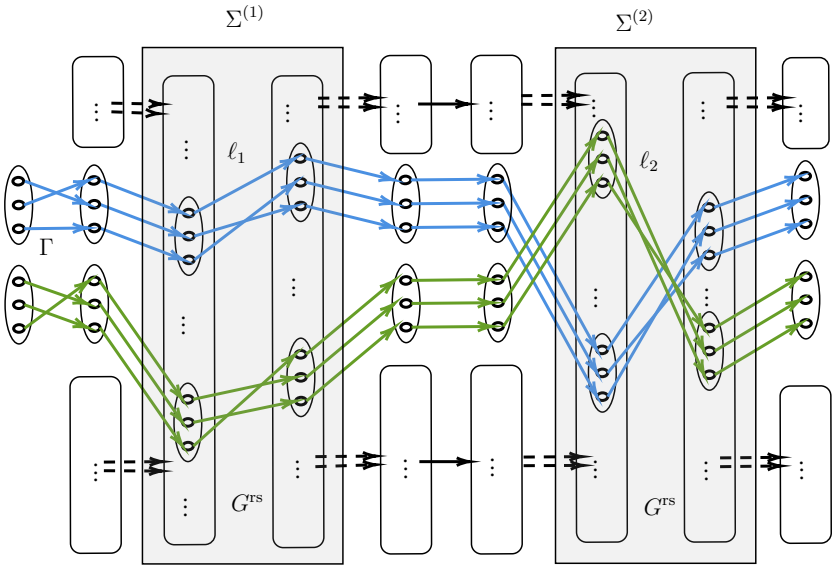


Figure 14: An illustration of a simple permutation hiding generator with $k = 2$. Compared to Figure 13, the basic permutation graph of $\text{JOIN}(\Gamma)$ is added to the left.

Now let us show that our construction possesses the required properties. We begin by proving that it is indeed a valid permutation graph for ρ with the required number of vertices.

Claim 6.19. *For any $\rho \in S_m^{\text{SIM}}$, the graph $\mathcal{G}^{\text{SIM}}(\rho)$ is a permutation graph for ρ with $n \leqslant 2\cdot 10^4 \cdot m/\alpha\beta$.*

Proof. We know that $\text{MULTI-BLOCK}(G_{\text{rs}}, \Sigma, L, M)$ is a permutation graph for $\text{JOIN}(\Gamma^*)$, where Γ^* is defined as in Problem 1 by Lemma 6.16. By Claim 6.1, the output $\mathcal{G}^{\text{SIM}}(\rho)$ is a permutation graph for ρ, as $\text{JOIN}(\Gamma^*) \circ \text{JOIN}(\Gamma) = \rho$ by construction.

The total number of vertices in MULTI-BLOCK$(G_{\text{rs}}, \Sigma, L, M)$ is $6k \cdot n^{\text{rs}} \cdot b$ by Claim 6.14. When we concatenate with BASIC(JOIN(Γ)), we add $2m$ vertices, so the total number is $6k \cdot n^{\text{rs}} \cdot b + 2m = 6 \cdot \frac{1600}{\beta} \cdot \frac{2m}{b\alpha} \cdot b + 2m \leqslant 2 \cdot 10^4 \cdot m/\alpha\beta$. ∎

Next, we will show that any 1-pass streaming algorithm can be run by the players of **Multi-HPH** and the referee on the graph $\mathcal{G}^{\text{SIM}}(\rho)$.

Claim 6.20. *For any $\rho \in S_m^{\text{SIM}}$ and given any 1-pass s-space streaming algorithm $\mathcal{A}$ using space at most s, the players and the referee of* **Multi-HPH** *(Problem 1) can run $\mathcal{A}$ on $\mathcal{G}^{\text{SIM}}(\rho)$ using at most s bits of communication per player.*

Proof. We know that the edges between layers $V^{6(i-1)+3}$ to $V^{6(i-1)+4}$ depend only on $\Sigma^{(i)}$ for $i \in [k]$ in MULTI-BLOCK$(G_{\text{rs}}, \Sigma, L, M)$ and all other edges are known to the referee, as they either depend on L, M or are fixed by Claim 6.15. The players and the referee run $\mathcal{A}$ on $G = \mathcal{G}^{\text{SIM}}(\rho)$ as follows.

(*i*) Player $Q^{(1)}$ runs $\mathcal{A}$ on the edges fixed by $\Sigma^{(1)}$ in G and writes the memory state on the board.

(*ii*) In increasing order of $i \in [k] \setminus \{1\}$, player $Q^{(i)}$ gets the memory state of $\mathcal{A}$ as written on the board by player $Q^{(i-1)}$ and continues to run $\mathcal{A}$ on the edges based on $\Sigma^{(i)}$ in G. Player $Q^{(i)}$, then writes the memory state back on the board.

(*iii*) The referee gets the state of $\mathcal{A}$ as written by $Q^{(k)}$, and adds all the other edges based on L, M and Γ to get the final output.

Each player writes on the board exactly once, and the communication per player is s bits. ∎

We will conclude this subsection by proving Lemma 6.18.

Proof of Lemma 6.18. The bound on the number of vertices follows readily from Claim 6.19. Let us assume that there is a 1-pass s-space streaming algorithm $\mathcal{A}$ which distinguishes between $\mathcal{G}^{\text{SIM}}(\rho_1)$ and $\mathcal{G}^{\text{SIM}}(\rho_2)$ for some $\rho_1, \rho_2 \in S_m^{\text{SIM}}$ with $s = o((m/b)^{1+2\beta/3})$ and advantage δ more than $(2m/b)^{-6}$. We will argue that it can be used to solve an instance of **Multi-HPH**.

Create an instance of **Multi-HPH**$_{r,t,b,k}$ with $\Gamma_{\text{Yes}} = \text{VEC}(\rho_1)$ and $\Gamma_{\text{No}} = \text{VEC}(\rho_2)$. We know from Claim 6.20 that the referee and players can run $\mathcal{A}$ using at most $s = o((m/b)^{1+2\beta/3}) = o(r^{1+2\beta/3})$ bits of communication per player. The total number of bits is $k \cdot s = O(1/\beta) \cdot o(r^{1+2\beta/3}) = o(rt)$, as $t = (r/\alpha)^\beta$. By Theorem 2, we know that the advantage the referee gains is at most

$$r \cdot O\left(\frac{k \cdot s}{r \cdot t}\right)^{k/32} = r \cdot o\left(\frac{r^{1+2\beta/3} \cdot \alpha^\beta}{r^{1+\beta} \cdot \beta}\right)^{50/\beta} = o(1/r^6) \cdot (\alpha^\beta \cdot \beta^{-1})^{50/\beta},$$

which is a contradiction. ∎

6.3 General Permutation Hiding in One Pass

In this subsection, we will hide any general permutation from 1-pass streaming algorithms by hiding multiple simple permutations constructed in Lemma 6.18 and prove Lemma 6.2. We first discuss how to divide any general $\rho \in S_m$ into multiple simple permutations with sorting networks.

The simple permutations we hid in Lemma 6.18 were all under the fixed partition Lex. It is easy to hide them even if they were under different partitions, as we will show.

Lemma 6.21. *Given any permutation $\rho \in S_m$ which is simple under partition $\mathcal{P}$ with m/b groups of size b, there is a simple permutation hiding generator for partition $\mathcal{P}$ with the same parameters as in Lemma 6.18.*

Proof. We permute the elements of $[m]$ based on $\mathcal{P}$ so that the partition will be Lex. To do so, let $f_i : P_i \to [b]$ be the lexicographic bijective mapping from group P_i to $[b]$ for $i \in [m/b]$, and let $g : [m] \to [m/b]$ be such that $g(j)$ denotes which group among the m/b groups $j \in [m]$ belongs to under $\mathcal{P}$. Define permutation $\mathsf{swap} \in S_m$ as,

$$\mathsf{swap}(j) = (g(j) - 1) \cdot b + f_{g(j)}(j)$$

for $j \in [m]$. Define permutation $\rho' \in S_m$ as,

$$\rho' = \mathsf{swap} \circ \rho \circ \mathsf{swap}^{-1}$$

Claim 6.22. *The permutation ρ' is simple under partition* Lex.

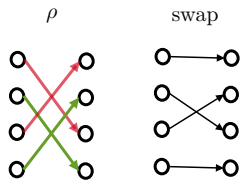
Proof. Let $h : [m] \to [m/b]$ be such that $h(j) = \left\lfloor \frac{j-1}{b} \right\rfloor + 1$. This denotes the group each element belongs to under partition Lex. We know that for any $x \in P_{g(x)}$, $\mathsf{swap}(x) \in \mathsf{Lex}_{g(x)}$, and for any $y \in \mathsf{Lex}_{h(y)}$, $\mathsf{swap}^{-1}(y) \in P_{h(y)}$ by the definition of swap. Thus, for any $y \in \mathsf{Lex}_{h(y)}$, we know that $\rho(\mathsf{swap}^{-1}(y)) \in P_{h(y)}$ as ρ is simple under $\mathcal{P}$. This implies that $\mathsf{swap}(\rho(\mathsf{swap}^{-1}(y))) \in \mathsf{Lex}_{h(y)}$. ∎

Now we can easily construct permutation graphs for ρ, by sampling $G \sim \mathcal{G}^{\text{SIM}}(\rho')$ under partition Lex from Lemma 6.18, and outputting

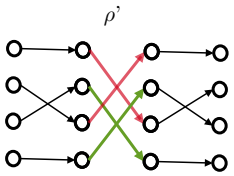
$$\textsc{Basic}(\mathsf{swap}) \circ G \circ \textsc{Basic}(\mathsf{swap}^{-1}).$$

We add $4m$ vertices to G, and the output is a permutation graph for ρ by Claim 6.1. ∎

Refer to Figure 15 for an illustration of the property proved by Lemma 6.21. Now we are ready to construct our generators.



(a) The permutation $\rho = (3, 4, 1, 2)$ under partition $\mathcal{P}$ with $P_1 = \{1, 3\}$ and $P_2 = \{2, 4\}$, and permutation $\mathsf{swap} = (1, 3, 2, 4)$ based on $\mathcal{P}, \mathsf{Lex}$.



(b) An illustration of permutation $\rho' = \mathsf{swap}^{-1} \circ \rho \circ \mathsf{swap} = (2, 1, 4, 3)$ which is simple under partition Lex.

Figure 15: An illustration of Lemma 6.21.

Construction of family $\mathcal{G}(m, n, 1, s, \delta) : S_m \to \mathcal{D}_m$ on input $\rho \in S_m$

(*i*) Get $d_{\text{SORT}} = c_{\text{SORT}} \cdot \log_b(m) = 100c_{\text{SORT}}/\beta$ simple permutations $\gamma_1, \gamma_2, \ldots, \gamma_{d_{\text{SORT}}} \in (S_m)$ under partitions $\mathcal{P}_1, \mathcal{P}_2, \ldots \mathcal{P}_{d_{\text{SORT}}}$ from Proposition 4.3 for $b = m^{\beta/100}$.

(*ii*) Sample $G_i \sim \mathcal{G}^{\text{SIM}}(\gamma_i)$ for $i \in [d_{\text{SORT}}]$ as in Lemma 6.21 and output $G = G_1 \circ G_2 \circ \ldots \circ G_{d_{\text{SORT}}}$.

Proof of Lemma 6.2. We know that G is a permutation graph for ρ by Lemma 6.21 and Claim 6.1. The total number of vertices in G is,

$$d_{\text{SORT}} \cdot 2 \cdot 10^4 \cdot m \cdot \frac{1}{\alpha\beta} = 2 \cdot c_{\text{SORT}} \cdot 10^6 \cdot m \cdot \frac{1}{\alpha\beta}.$$

Let $\mathcal{A}$ be a 1-pass streaming algorithm that distinguishes between graphs sampled from $\mathcal{G}(\rho_1)$ and $\mathcal{G}(\rho_2)$ using space $s = o(m^{1+\beta/2})$ and advantage more than m^{-5}. That is,

$$\|\text{MEM}^1_{\mathcal{A}}(\mathcal{G}(\rho_1)) - \text{MEM}^1_{\mathcal{A}}(\mathcal{G}(\rho_2))\|_{\text{tvd}} \geqslant m^{-5}.$$

Let $\gamma_{1,i}, \gamma_{2,i}$ be the simple permutations under partition $\mathcal{P}_i$ from step (i) of the construction for $i \in [d_{\text{SORT}}]$ for ρ_1, ρ_2 respectively. For any $\rho \in S_m$ distribution $\mathcal{G}(\rho)$ is fixed by sampling from $\mathcal{G}^{\text{SIM}}(\gamma_i)$ for all $i \in [d_{\text{SORT}}]$. By Fact A.11, we know that,

$$\begin{aligned}
&\|\text{MEM}(\mathcal{G}(\rho_1)) - \text{MEM}(\mathcal{G}(\rho_2))\|_{\text{tvd}} \\
&\leqslant \|\text{MEM}(\mathcal{G}^{\text{SIM}}(\gamma_{1,1}), \ldots, \mathcal{G}^{\text{SIM}}(\gamma_{1,d_{\text{SORT}}})) - \text{MEM}(\mathcal{G}^{\text{SIM}}(\gamma_{2,1}), \ldots, \mathcal{G}^{\text{SIM}}(\gamma_{2,d_{\text{SORT}}}))\|_{\text{tvd}} \\
&\leqslant \sum_{i=1}^{d_{\text{SORT}}} \|\text{MEM}(\mathcal{G}^{\text{SIM}}(\gamma_{1,i})) - \text{MEM}(\mathcal{G}^{\text{SIM}}(\gamma_{2,i}))\|_{\text{tvd}} \qquad \text{(by the hybrid argument Proposition 4.7)} \\
&\leqslant d_{\text{SORT}} \cdot (r)^{-6} (\alpha^\beta \cdot \beta^{-1})^{50/\beta} \leqslant (100 c_{\text{SORT}}/\beta) \cdot m^{-5} \cdot (\alpha^\beta \cdot \beta^{-1})^{50/\beta},
\end{aligned}$$

where in the last inequality, we have used Lemma 6.18 because algorithm $\mathcal{A}$ uses $s = o(m^{1+\beta/2}) = o(\left(\frac{2m}{b}\right)^{1+2\beta/3})$ space. The proof follows as for our choice of $b = m^{\beta/100}$, $r^{-6} < m^{-5}$. ∎

7 Multi-pass Permutation Hiding

In this section, we will construct permutation hiding graphs against p-pass semi-streaming algorithms for any general integer $p \geqslant 1$ with induction. We restate Theorem 3 here and prove it in this section.

Theorem (Restatement of Theorem 3)**.** *There exists a permutation hiding generator $\mathcal{G}(m, n, p, s, \delta)$ where p is any positive integer with the following parameters:*

- *Space $s = o(m^{1+\beta/2})$,*
- *Number of vertices $n = \Theta(1/\alpha\beta^2)^p \cdot m$,*
- *Error parameter $\delta = (p/\beta)^{\Theta(1)/\beta} \cdot \Theta(1/\beta)^{2p} \cdot 1/\text{poly}(m)$.*

Our overall strategy is the same as that of Lemma 6.2, with the key difference being that we hide simple permutations from p-pass streaming algorithms instead by using permutation hiding generators for $p-1$-pass streaming algorithms. Our inductive hypothesis is as follows.

Assumption 7.1 (Inductive Hypothesis). *When considering p pass algorithms, there exists a permutation hiding generator $\mathcal{G}(m, n, p-1, s, \delta)$ for the following parameters:*

$$\begin{aligned}
n &= \left(\frac{2.5 \cdot c_{\text{SORT}} \cdot 10^6}{\alpha \cdot \beta^2}\right)^{p-1} \cdot m; && \text{(number of vertices)} \\
s &:= o(m^{1+\beta/2}); && \text{(space of streaming algorithm)} \\
\delta &:= ((p-1)/\beta)^{50/\beta} \cdot ((4 \cdot 10^6 \cdot c_{\text{SORT}}/\beta^2)^{(p-1)}) \cdot m^{-5} \cdot \alpha., \\
& && \text{(probability of success of the algorithm)}
\end{aligned}$$

Notation. We let $\mathcal{G}_{p-1}$ refer to the permutation hiding generator in Assumption 7.1. We use $N_{p-1}(m)$ and $\Delta_{p-1}(m)$ to denote the number of vertices and the probability of success for $\mathcal{G}_{p-1}$ respectively when the size of the permutation is m.

The first step again is to hide simple permutations. For the rest of this section, we will define some more constructs to hide simple permutations, and then prove Theorem 3.

7.1 Building Blocks for Multi-Pass Hiding

In this subsection, we will adapt the definitions of blocks and multi-blocks based on the number of passes of the streaming algorithm (denoted by p) which we want to guard against.

First, we want to extend any permutation $\sigma \in S_{n^{\text{rs}}}$ to $S_{n^{\text{rs}} \cdot b}$ in a specific way.

Definition 7.2 (Extended Permutation)**.** Given a permutation $\sigma \in S_{n^{\text{rs}}}$, and an integer b, the extended permutation $\sigma' \in S_{n^{\text{rs}} \cdot b}$, denoted by $\textsc{Extend}(\sigma, b)$ is defined as,

$$\sigma'((x-1) \cdot b + j) = (\sigma(x) - 1) \cdot b + j$$

for all $x \in [n^{\text{rs}}], j \in [b-1]$.

Informally, we picked the lexicographic partition of $[n^{\text{rs}} \cdot b]$ into n^{rs} groups of size b, and used σ to permute the groups among each other. Permutation σ' does not permute the elements within each of the groups. The following observation connects extended permutations to one of the primitives we constructed in Section 6.1.

Observation 7.3. *For any $\sigma \in S_{n^{\text{rs}}}$ and integer $b \geqslant 1$, the graph $G = \textsc{Permute}(\sigma, b)$ from Definition 6.5 is a permutation graph for $\textsc{Extend}(\sigma, b) \in S_{n^{\text{rs}} \cdot b}$.*

Proof. Let $\rho = \textsc{Extend}(\sigma, b) \in S_{n^{\text{rs}} \cdot b}$. For any $i \in [n^{\text{rs}}], j \in [b]$ element $(i-1)b + j \in [n^{\text{rs}} \cdot b]$, we know that $\rho((i-1)b + j) = \sigma(i) + j$. In the graph $\textsc{Permute}(\sigma, b)$ (with layers V^1 and V^2), the vertex identified by $(i-1)b + j$ in V^1 is connected to $(\sigma(i) - 1)b + j$ by definition. ∎

Definition 7.4 (p-Pass Block Distribution)**.** For any $r, t, b \geqslant 1$, and inputs (r,t)-RS graph G_{rs}, permutation matrix $\Sigma \in (S_b)^{t \times r}$, index $\ell \in [t]$ and edge tuple $E^* = (e_1, e_2, \ldots e_{r/2})$, let:

$$(\sigma_{\text{L}}, \sigma_{\text{R}}) = \textsc{Edge-Pick}(G_{\text{rs}}, \ell, E^*).$$

be permutations in $S_{n^{\text{rs}}}$ as similar to Definition 6.9. We extend them to $\sigma_1, \sigma_2 \in S_{n^{\text{rs}} \cdot b}$ as:

$$\sigma_1 = \textsc{Extend}(\sigma_{\text{L}}, b) \qquad \text{and} \qquad \sigma_2 = \textsc{Extend}(\sigma_{\text{R}}, b).$$

We define the **p-pass block distribution**, denoted by $\mathsf{p\text{-}Block}(G_{\text{rs}}, \Sigma, j^*, E^*)$ as follows. (See Figure 16 for an illustration.)

(i) Let G^{Σ} be the encoded RS-graph of G_{rs} on Σ from Definition 6.6.

(ii) Sample $G_L \sim \mathcal{G}_{p-1}(\sigma_1)$ and $G_R \sim \mathcal{G}_{p-1}(\sigma_2)$ from Assumption 7.1.

(iii) Output the graph $G = G_R \circ G^{\Sigma} \circ G_L$.

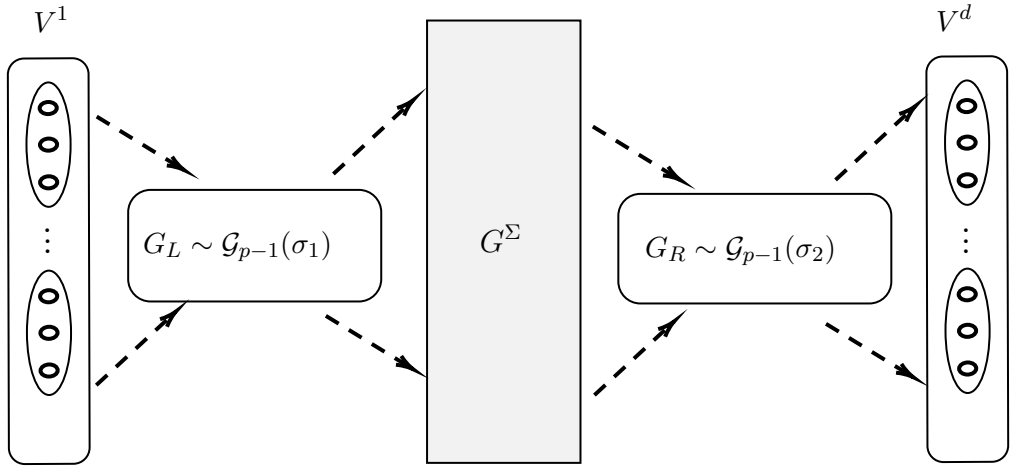


Figure 16: An illustration of a graph sampled from $\mathsf{p\text{-}Block}(G_{\text{rs}}, \Sigma, j^*, E^*)$.

Let us compare Definition 7.4 to that of blocks in Definition 6.9. In a block, we concatenate basic permutation graphs for $\textsc{Extend}(\sigma_{\text{L}}, b)$ and $\textsc{Extend}(\sigma_{\text{R}}, b)$ to either side of the encoded RS-graph based on Σ, by Observation 7.3. However, in a p-pass block distribution, we *hide* the two permutations $\textsc{Extend}(\sigma_{\text{L}}, b)$ and $\textsc{Extend}(\sigma_{\text{R}}, b)$ from $S_{n^{\text{rs}}\cdot b}$ from $p-1$ pass streaming algorithms inductively. This is the key change that allows us to hide our permutations from p-pass algorithms.

Let us show that graphs sampled from p-pass block distributions are also valid permutation graphs for some specific permutations, similar to blocks.

Claim 7.5. *Any graph $G \sim \mathsf{p\text{-}Block}(G_{\text{rs}}, \Sigma, j^*, E^*)$ is a permutation graph for $\rho \in S_m$ defined by Eq (22) with $2n^{\text{rs}} \cdot b + 2N_{p-1}(n^{\text{rs}} \cdot b)$ vertices.*

Proof. For any graph G sampled from $\mathsf{p\text{-}Block}(G_{\text{rs}}, \Sigma, j^*, E^*)$, Claim 6.12 applies because G is also a concatenation of permutation graphs of the same permutations, by Observation 7.3. The bound on the total number of vertices follows from Definition 6.6 and Assumption 7.1. ∎

We can show that two different p-pass block distributions sharing the input Σ cannot be distinguished by $(p-1)$-pass streaming algorithms with a high advantage, based on the guarantees in Assumption 7.1.

Claim 7.6. *For any $\ell_1, \ell_2 \in [t]$ and two edge tuples E_1^*, E_2^* from $M_{\ell_1}^{\text{rs}}$ and $M_{\ell_2}^{\text{rs}}$ respectively, the two distributions $\mathcal{D}_1 = \mathsf{p\text{-}Block}(G_{\text{rs}}, \Sigma, \ell_1, E_1^*)$ and $\mathcal{D}_2 = \mathsf{p\text{-}Block}(G_{\text{rs}}, \Sigma, \ell_2, E_2^*)$ are $2\Delta_{p-1}(n^{\text{rs}} \cdot b)$-indistinguishable for $(p-1)$-pass streaming algorithms using space at most $o((n^{\text{rs}} \cdot b)^{1+\beta/2})$.*

Proof. Let A be a $p-1$ pass streaming algorithm using space $o((n^{\text{rs}} \cdot b)^{1+\beta/2})$. Define the following permutations:

$$(\sigma_{\text{L}}^1, \sigma_{\text{R}}^1) = \textsc{Edge-Pick}(G_{\text{rs}}, \ell_1, E_1^*) \qquad (\sigma_{\text{L}}^2, \sigma_{\text{R}}^2) = \textsc{Edge-Pick}(G_{\text{rs}}, \ell_2, E_2^*)$$
$$\tau_{1,1} = \textsc{Extend}(\sigma_{\text{L}}^1, b) \qquad \tau_{2,1} = \textsc{Extend}(\sigma_{\text{L}}^2, b)$$
$$\tau_{1,2} = \textsc{Extend}(\sigma_{\text{R}}^1, b) \qquad \tau_{2,2} = \textsc{Extend}(\sigma_{\text{R}}^2, b)$$

where $\tau_{1,1}, \tau_{1,2}, \tau_{2,1}$ and $\tau_{2,2}$ belong to $S_{n^{\text{rs}}\cdot b}$. Distribution $\mathcal{D}_i$ is fixed based on samples from $\mathcal{G}_{p-1}(\tau_{i,1})$ and $\mathcal{G}_{p-1}(\tau_{i,2})$ for $i = 1, 2$. By Fact A.11, we have that, .

$$\|\textsc{mem}_A^{p-1}(\mathcal{D}_1) - \textsc{mem}_A^{p-1}(\mathcal{D}_2)\|_{\text{tvd}} \leqslant \|(\mathcal{G}_{p-1}(\tau_{1,1}), \mathcal{G}_{p-1}(\tau_{1,2})) - (\mathcal{G}_{p-1}(\tau_{2,1}), \mathcal{G}_{p-1}(\tau_{2,2}))\|_{\text{tvd}}$$

$$\leqslant \sum_{j=1,2} \|\mathcal{G}_{p-1}(\tau_{1,j}) - \mathcal{G}_{p-1}(\tau_{2,j})\|_{\text{tvd}} \qquad \text{(by Proposition 4.7)}$$

$$\leqslant 2\Delta_{p-1}(n^{\text{rs}} \cdot b). \qquad \blacksquare \qquad \text{(by Assumption 7.1)}$$

We can extend Definition 6.13 to distributed p-pass multi-blocks as follows.

Definition 7.7 (p-pass Multi-Block Distribution)**.** Given inputs $G_{\text{rs}}, \Sigma \in (S_b)^{t\times r}, L \in [t]^k$ a hypermatching $M \subset [r]^k$ of size $r/2$ with E_i^* defined similar to Definition 6.13 for $i \in [k]$, the p-pass multi-block distribution, denoted by $\mathsf{p\text{-}Multi\text{-}Block}(G_{\text{rs}}, \Sigma, L, M)$ is defined as,

(i) For each $i \in [k]$, sample graph $G_i \sim \mathsf{p\text{-}Block}(G_{\text{rs}}, \Sigma^{(i)}, \ell_i, E_i^*)$.

(ii) Output $G_1 \circ G_2 \circ \ldots \circ G_k$.

Next, we show that graphs sampled from multi-block distributions are valid permutation graphs.

Claim 7.8. *Any graph sampled from* $\mathsf{p\text{-}Multi\text{-}Block}(G_{\text{rs}}, \Sigma, L, M)$ *is a permutation graph for* $\gamma^\star$ *defined by Eq (24) with* $k(2n^{\text{rs}} \cdot b + 2N_{p-1}(n^{\text{rs}} \cdot b))$ *vertices.*

Proof. Any graph sampled from $\mathsf{p\text{-}Multi\text{-}Block}(G_{\text{rs}}, \Sigma, L, M, \Gamma)$ is a permutation graph for $\gamma^\star$ by Claim 6.1 and Claim 7.5. As we concatenate k graphs sampled from p-pass block distributions, the total number of vertices follows from Claim 7.5. $\blacksquare$

Next, we will prove the analog of Claim 7.6 for multi-blocks, with a slightly larger advantage.

Claim 7.9. *For any* $L_1, L_2 \in [t]^k$ *and two hypermatchings* M_1, M_2*, the two distributions*

$$\mathcal{D}_1 = \mathsf{p\text{-}Multi\text{-}Block}(G_{\text{rs}}, \Sigma, L_1, M_1) \quad \textit{and} \quad \mathcal{D}_2 = \mathsf{p\text{-}Multi\text{-}Block}(G_{\text{rs}}, \Sigma, L_2, M_2)$$

are $2k \cdot \Delta_{p-1}(n^{\text{rs}} \cdot b)$*-indistinguishable for* $(p-1)$*-pass streaming algorithms* $o((n^{\text{rs}} \cdot b)^{1+\beta/2})$ *space.*

Proof. Let A be a $p-1$ pass algorithm using space $o((n^{\text{rs}} \cdot b)^{1+\beta/2})$. For $i \in [k]$, let $j_{1,i}^*, j_{2,i}^*$ and $E_{1,i}^*, E_{2,i}^*$ be the corresponding values of j_i^*, E_i^* from Definition 7.7 on inputs $G_{\text{rs}}, \Sigma, L_1, M_1$ and $G_{\text{rs}}, \Sigma, L_2, M_2$ respectively. Let the distribution $\mathsf{p\text{-}Block}(G_{\text{rs}}, \Sigma, j_{j,i}^*, E_{j,i}^*)$ be referred to by $\mathcal{B}_{j,i}$ for $j = 1, 2$ and $i \in [k]$. For $j = 1, 2$, distribution $\mathcal{D}_j$ is fixed based on all samples from $\mathcal{B}_{j,i}$ for $i \in [k]$. By Fact A.11, we have that, .

$$\|\text{MEM}_A^{p-1}(\mathcal{D}_1) - \text{MEM}_A^{p-1}(\mathcal{D}_2)\|_{\text{tvd}} \leqslant \|(\mathcal{B}_{1,1}, \ldots, \mathcal{B}_{1,k}) - (\mathcal{B}_{2,1}, \ldots, \mathcal{B}_{2,k})\|_{\text{tvd}}$$

$$\leqslant \sum_{i\in[k]} \|\mathcal{B}_{1,i} - \mathcal{B}_{2,i}\|_{\text{tvd}} \quad \text{(by hybrid argument Proposition 4.7)}$$

$$\leqslant 2k\Delta_{p-1}(n^{\text{rs}} \cdot b). \qquad \blacksquare \qquad \text{(by Assumption 7.1)}$$

7.2 Permutation Hiding in Multiple Passes

Our approach is similar to the proof of Lemma 6.2, where we hide multiple simple permutations to hide any σ from S_m. Let us show that simple permutations under partition Lex can be hidden from p-pass algorithms.

Lemma 7.10. *Under Assumption 7.1, there exists a simple permutation hiding generator for partition* Lex*,* $\mathcal{G}^{\text{SIM}} : S^{\text{SIM}} \to \mathcal{D}_m$ *for* p*-pass streaming algorithms using space* $s = o((m/b)^{1+2\beta/3})$ *when* $b = m^{\beta/100}$ *with the following parameters:*

(*i*) *The total number of vertices* $n = (3200/\beta) \cdot (n^{\text{rs}} \cdot b + N_{p-1}(n^{\text{rs}} \cdot b)) + N_{p-1}(m)$.

(*ii*) *The advantage gained is at most* $(3200/\beta) \cdot \Delta_{p-1}(n^{\text{rs}} \cdot b) + \Delta_{p-1}(m) + \alpha \cdot m^{-5} \cdot (p/\beta)^{50/\beta}$.

The explicit construction of simple permutation hiding generators for p passes follows.

Construction of $\mathcal{G}^{\text{SIM}}(n, p, s, \mathsf{Lex}, \delta) : S^{\text{SIM}} \to \mathcal{D}_m$ (denoted by $\mathcal{G}_p^{\text{SIM}}$) on input $\rho \in S^{\text{SIM}}$.

(*i*) Fix an (r, t)-RS graph G_{rs} with $r = 2m/b = n^{\text{rs}}/\alpha$ and $t = (n^{\text{rs}})^{\beta}$.

(*ii*) Instantiate k players and the referee in **Multi-HPH**$_{r,t,b,k}$ (Problem 1) with $k = 1600/\beta$ such that the solution to the instance, $\Gamma^* \circ \Gamma$ is $\text{VEC}(\rho)$. Let the variables in the instance be Σ, L, M, and Γ.

(*iii*) Sample graph $G^{\Sigma} \sim \mathsf{p\text{-}Multi\text{-}Block}(G_{\text{rs}}, \Sigma, L, M)$ and $G^{\Gamma} \sim \mathcal{G}_{p-1}(\text{JOIN}(\Gamma))$.

(*iv*) Output $G^{\Sigma} \circ G^{\Gamma}$.

First, let us show that any graph output by the distribution $\mathcal{G}^{\text{SIM}}$ for p-pass algorithms is a valid permutation graph.

Claim 7.11. *Given any $\rho \in S_m^{\text{SIM}}$, any graph $G \sim \mathcal{G}_p^{\text{SIM}}(\rho)$ is a permutation graph for ρ with $n = 2k(N_{p-1}(n^{\text{rs}} \cdot b) + n^{\text{rs}} \cdot b) + N_{p-1}(m)$ vertices.*

Proof. We know that $\mathsf{p\text{-}Multi\text{-}Block}(G_{\text{rs}}, \Sigma, L, M)$ is a permutation graph for $\text{JOIN}(\Gamma^*)$, where Γ^* is defined as in Problem 1 by Claim 7.8. $\mathcal{G}_p^{\text{SIM}}(\rho)$ is a permutation graph for ρ by Claim 6.1.

The bound on the total number of vertices in $\mathsf{p\text{-}Multi\text{-}Block}(G_{\text{rs}}, \Sigma, L, M)$ follows from Claim 7.8 and Assumption 7.1. ∎

To prove that the preceding construction is a valid simple permutation hiding generator for p-pass algorithms, we have to argue that no p-pass algorithm using space $s = o((m/b)^{1+2\beta/3})$ can distinguish between two graphs output from different distributions. Let A be one such algorithm which distinguishes between $\mathcal{G}(\rho_1)$ and $\mathcal{G}(\rho_2)$ for some $\rho_1, \rho_2 \in S_m^{\text{SIM}}$. We can assume A is deterministic by Yao's minimax principle.

Here our approach will differ from the 1-pass case. To run streaming algorithm A on the input graph for p-passes the naive way, back and forth communication between the referee and the players is required, and this is not possible. Instead, the players will run algorithm A for $p - 1$ passes on a *different graph*, assuming some input on behalf of the referee, and the last pass is run on the original graph. We will show that this will be sufficient to solve **Multi-HPH**. Before giving the protocol to run p-passes of A, let us see that one pass of A can be run using the inputs of both the referee and the players.

Claim 7.12. *Suppose we are given an instance of* **Multi-HPH** *with inputs $G_{\text{rs}}, \Sigma, L, M$ and Γ. Let $G_1 \sim \mathsf{p\text{-}Multi\text{-}Block}(G_{\text{rs}}, \Sigma, L, M)$ and let $G_2 \sim \mathcal{G}_{p-1}(\text{JOIN}(\Gamma))$. The players and the referee can run any streaming algorithm A using space at most s for one pass on graph $G_1 \circ G_2$ using at most $k \cdot s$ bits of communication.*

Proof. For $i \in [k]$, in any graph $G_i \sim \mathsf{p\text{-}Block}(G_{\text{rs}}, \Sigma^{(i)}, \ell_i, E_i^*)$, the edges based on $\Sigma^{(i)}$ can be added by player $Q^{(i)}$ and the other edges based on L, M and E_i^* can be added by the referee. No edge depends on both the inputs from the player and the referee.

Each player $Q^{(i)}$ in increasing order of $i \in [k]$, takes the memory state of A after adding all the edges based on $\Sigma^{(1)}, \Sigma^{(2)}, \ldots, \Sigma^{(i-1)}$, runs A on the edges based on $\Sigma^{(i)}$ and then uses s bits of communication to write the new memory state of A on the board. This is totally $k \cdot s$ bits of communication. The referee can use the state from $Q^{(k)}$ and add the edges based on L, M, and Γ, which concludes the proof. ∎

Let us see how the players and the referee can *simulate* algorithm A for p passes.

Protocol Π_A for Multi-HPH using the p-pass streaming algorithm A

(*i*) The players $Q^{(i)}$ for $i \in [k]$ pick the lexicographically first $\overline{L} \in [t]^k$, $\overline{\Gamma} \in (S_b)^{r/2}$ and a hypermatching $\overline{M}$ from $[r]^k$ collectively.

(*ii*) Sample a graph $\overline{G_\Sigma} \sim \mathsf{p\text{-}Multi\text{-}Block}(G_{\text{rs}}, \Sigma, \overline{L}, \overline{M})$ and $\overline{G_\Gamma} \sim \mathcal{G}_{p-1}(\textsc{join}(\overline{\Gamma}))$.

(*iii*) Run algorithm A for $p-1$ passes on $\overline{G} = \overline{G_\Sigma} \circ \overline{G_\Gamma}$.

(*iv*) The players and the referee jointly sample $G_\Sigma \sim \mathsf{p\text{-}Multi\text{-}Block}(G_{\text{rs}}, \Sigma, L, M)$, and the referee samples $G_\Gamma \sim \mathcal{G}_{p-1}(\textsc{join}(\Gamma))$.

(*v*) In the last pass, the players and the referee run algorithm A on $G = G_\Sigma \circ G_\Gamma$.

Let us argue that running such a protocol Π_A is possible for any p pass streaming algorithm A.

Claim 7.13. *There is a way for the players and the referee to run protocol Π_A using at most $k \cdot p \cdot s$ total communication.*

Proof. The values of $\overline{\Gamma}, \overline{L}$ and $\overline{M}$ are known to all players $Q^{(i)}$ for $i \in [k]$. The distribution in Assumption 7.1 is known to all the players too. Let $\overline{L} = (\overline{\ell_1}, \overline{\ell_2}, \ldots, \overline{\ell_k})$, and let edge tuple $\overline{E_i^*} = (\overline{M}_{1,i}, \overline{M}_{2,i}, \ldots, \overline{M}_{r/2,i})$ be $r/2$ edges in $M^{\text{rs}}_{\overline{\ell_i}}$ for $i \in [k]$.

In Definition 7.7, we know that graphs are sampled from distribution $\mathsf{p\text{-}Block}(G_{\text{rs}}, \Sigma^{(i)}, \overline{\ell_i}, \overline{E_i^*})$ for each $i \in [k]$. The graph G_{rs}, index $\overline{\ell_i}$ and edges $\overline{E_i^*}$ for $i \in [k]$ are known to all the players since they fix graph G_{rs}. Hence, player $Q^{(i)}$ can sample a graph from $\overline{G_i} \sim \mathsf{p\text{-}Block}(G_{\text{rs}}, \Sigma^{(i)}, \overline{\ell_i}, \overline{E_i^*})$ for each $i \in [k]$ privately.

Sampling from $\mathcal{G}_{p-1}(\overline{\Gamma})$ can be done by any player since they all know what $\overline{\Gamma}$ is. To execute 1 pass of step (*iii*) of Π_A on graph $\overline{G}$, the player $Q^{(i)}$ can, in order, add the edges from $\overline{G_i}$ for $i \in [k]$ and then any player can add the edges from $\overline{G_\Gamma}$. The players and the referee can execute step (*iv*) using Claim 7.12. ∎

Now we argue the correctness of this protocol. We need some extra notation before we proceed.

Notation. We fix the input of the **Multi-HPH**$_{r,t,b,k}$ instance to be $G_{\text{rs}}, \Sigma, L, M$ and Γ. Let δ_A be the advantage gained by p-pass streaming algorithm A that distinguishes between the two distributions $\mathcal{G}_p^{\text{SIM}}(\rho_1)$ and $\mathcal{G}_p^{\text{SIM}}(\rho_2)$. Let $s_A = o((m/b)^{1+2\beta/3})$ denote the space used by A.

Let Π^{fake} be the protocol of running A for p-passes on graph G which is the concatenation of graphs sampled from distributions $\mathsf{p\text{-}Multi\text{-}Block}(G_{\text{rs}}, \Sigma, L, M)$ and $\mathcal{G}_{p-1}(\textsc{join}(\Gamma))$. (Note that Π^{fake} is an impossible protocol to run because there is no back and forth communication between the referee and the players; however, it is a well-defined random variable/distribution.)

Let $\textsc{mem}_A^j(\Pi^{\mathsf{fake}})$ be the random variable denoting the memory state of A after running j passes of G in Π^{fake} for $j \leqslant p$. Let $\textsc{mem}_A^j(\Pi_A)$ be the random variable denoting the memory contents of the board after running j passes of Π_A on graph $\overline{G}$ for $j \leqslant p-1$ and graph G for pass p.

Let graph G^{fixed} denote the subgraph of G with all the edges added by the players $Q^{(i)}$ for $i \in [k]$ based on $\Sigma^{(i)}$. This graph is fixed because we have fixed the input to $\mathsf{MPH}_{t,r,b,k}$. Let $\mathcal{G}^{\mathsf{rest}}$ be the distribution of the rest of the edges in G.

Claim 7.14. *Protocol Π_A solves* **Multi-HPH**$_{r,t,b,k}$ *with advantage at least*

$$\delta_A - (2k \cdot \Delta_{p-1}(n^{\text{rs}} \cdot b) + \Delta_{p-1}(m))$$

and total communication at most $k \cdot p \cdot s_A$ when $s_A = o(m^{1+\beta/2})$.

Proof. The contents of the last pass are a deterministic function of the contents of pass $p-1$ and the edges added. By Fact A.11, we have that,

$$\|\textsc{mem}_A^p(\Pi_A) - \textsc{mem}_A^p(\Pi^{\mathsf{fake}})\|_{\text{tvd}} \leqslant \|(\textsc{mem}_A^{p-1}(\Pi_A), G^{\mathsf{fixed}}, \mathcal{G}^{\mathsf{rest}}) - (\textsc{mem}_A^{p-1}(\Pi^{\mathsf{fake}}), G^{\mathsf{fixed}}, \mathcal{G}^{\mathsf{rest}})\|_{\text{tvd}}.$$

The edges added in the last pass for both protocols are the ones from G^{fixed} which are fixed, and the others from distribution $\mathcal{G}^{\mathsf{rest}}$. We can write the RHS term as,

$$\begin{aligned}
&\|(\textsc{mem}_A^{p-1}(\Pi_A), G^{\mathsf{fixed}}, \mathcal{G}^{\mathsf{rest}}) - (\textsc{mem}_{\Pi^{\mathsf{fake}}}^{p-1}, G^{\mathsf{fixed}}, \mathcal{G}^{\mathsf{rest}})\|_{\text{tvd}} \\
&\quad \leqslant \|\textsc{mem}_A^{p-1}(\Pi_A) - \textsc{mem}_A^{p-1}(\Pi^{\mathsf{fake}})\|_{\text{tvd}} \\
&\qquad + \mathop{\mathbb{E}}_{\pi \sim \textsc{mem}_A^{p-1}(\Pi_A)} \left(\| \left((G^{\mathsf{fixed}}, \mathcal{G}^{\mathsf{rest}}) \mid \textsc{mem}_A^{p-1}(\Pi_A) = \pi \right) - \left((G^{\mathsf{fixed}}, \mathcal{G}^{\mathsf{rest}}) \mid \textsc{mem}_A^{p-1}(\Pi^{\mathsf{fake}}) = \pi \right) \|_{\text{tvd}} \right) \\
&\qquad\qquad \text{(by Fact A.9)} \\
&\quad \leqslant \|\textsc{mem}_A^{p-1}(\Pi_A) - \textsc{mem}_A^{p-1}(\Pi^{\mathsf{fake}})\|_{\text{tvd}},
\end{aligned}$$

where in the last step, the second term becomes zero, as they are the exact same distribution. For the first $p-1$ rounds, Π_A and Π^{fake} run algorithm A on input graphs sampled from $\mathsf{p\text{-}Multi\text{-}Block}(G_{\text{rs}}, \Sigma, \overline{L}, \overline{M}), \mathcal{G}_{p-1}(\textsc{join}(\overline{\Gamma}))$ and $\mathsf{p\text{-}Multi\text{-}Block}(G_{\text{rs}}, \Sigma, L, M), \mathcal{G}_{p-1}(\textsc{join}(\Gamma))$ respectively. As this is a $p-1$ pass streaming algorithm using space $o(m^{1+\beta/2})$, by Claim 7.9 and Assumption 7.1,

$$\|\textsc{mem}_A^{p-1}(\Pi_A) - \textsc{mem}_A^{p-1}(\Pi^{\mathsf{fake}})\|_{\text{tvd}} \leqslant 2k \cdot \Delta_{p-1}(n^{\text{rs}} \cdot b) + \Delta_{p-1}(m).$$

For any input, by our assumption, we know that protocol Π^{fake} distinguishes between $\tilde{\mathcal{G}}(\Gamma_1)$ and $\tilde{\mathcal{G}}(\Gamma_2)$ with advantage at least δ_A. However, we have shown that for any input, the total variation distance between the transcripts of protocols Π^{fake} and Π_A is low. Hence,

$$\begin{aligned}
\Pr\left(\text{Success of } \Pi^{\mathsf{fake}}\right) &\leqslant \Pr\left(\text{Success of } \Pi_A\right) + \|\textsc{mem}_A^p(\Pi_A) - \textsc{mem}_A^p(\Pi^{\mathsf{fake}})\|_{\text{tvd}} \\
\Pr\left(\text{Success of } \Pi_A\right) &\geqslant \frac{1}{2} + \delta_A - (2k \cdot \Delta_{p-1}(n^{\text{rs}} \cdot b) + \Delta_{p-1}(m)). \qquad \blacksquare
\end{aligned}$$

We have all we need to prove Lemma 7.10 now.

Proof of Lemma 7.10. The total number of vertices in our construction follows directly from Claim 7.11.

By Claim 7.13 and Claim 7.14, we know that Π_A can solve instances of $\mathsf{MPH}_{t,r,b,k}$ with advantage at least $\delta_A - (2k \cdot \Delta_{p-1}(n^{\text{rs}} \cdot b) + \Delta_{p-1}(rb/2))$ with total communication $k \cdot p \cdot s_A$.

When $p = o(\beta \cdot r^{\beta/3})$, we have that $k \cdot p \cdot s_A = O(1/\beta) \cdot o((m/b)^{1+2\beta/3}) \cdot o(\beta \cdot r^{\beta/3}) = o(rt)$. Hence Theorem 2 is applicable. We know that the advantage gained by Π_A is at most,

$$r \cdot o\left(\frac{k \cdot p \cdot s_A}{r \cdot t}\right)^{k/32} \leqslant r \cdot o\left(\frac{\alpha^\beta \cdot k \cdot p \cdot r^{1+2\beta/3}}{r^{1+\beta}}\right)^{50/\beta} \leqslant \frac{1}{r^{15}} \cdot (\alpha^\beta \cdot \beta^{-1} \cdot p)^{50/\beta}$$

Algorithm A is given $o(r^{1+2\beta/3}) = o((m)^{1+\beta/2})$ when $b = m^{\beta/100}$. So we can lower bound the advantage of Π_A from Claim 7.14 as,

$$\delta_A \leqslant (2k \cdot \Delta_{p-1}(n^{\mathrm{rs}} \cdot b) + \Delta_{p-1}(m)) + m^{-5} \cdot (p \cdot \beta^{-1})^{50/\beta} \cdot \alpha. \qquad \blacksquare$$

Using Lemma 7.10, we can construct the permutation hiding generator for p-passes similar to the construction in Section 6 using simple permutation hiding generators for p passes.

Proof of Theorem 3. The base case was proved in Lemma 6.2. We use Assumption 7.1 to infer the statement of the theorem, completing the proof by induction. We fix $b = m^{\beta/100}$, and we concatenate $c_{\mathrm{SORT}} \cdot 100/\beta$ graphs sampled from the simple permutation hiding generator for p-pass algorithms. Using Lemma 7.10, we can bound the total number of vertices as,

$$\begin{aligned} n &\leqslant d_{\mathrm{SORT}} \cdot ((3200/\beta) \cdot (n^{\mathrm{rs}} \cdot b + N_{p-1}(n^{\mathrm{rs}} \cdot b)) + N_{p-1}(m)) \\ &\leqslant 3.2 \cdot 10^5 \cdot c_{\mathrm{SORT}} \cdot \frac{1}{\beta^2} \cdot \left(\frac{2.5 \cdot c_{\mathrm{SORT}} \cdot 10^6}{\alpha\beta^2}\right)^{p-1} \cdot (6m/\alpha) \\ &\leqslant \left(\frac{2.5 \cdot c_{\mathrm{SORT}} \cdot 10^6}{\alpha\beta^2}\right)^{p} \cdot m. \end{aligned}$$

Using the bound on the advantage in Lemma 7.10, we get that, for any $\rho_1, \rho_2 \in S_m$,

$$\begin{aligned} &\|\mathrm{MEM}_A^p(\mathcal{G}(\rho_1)) - \mathrm{MEM}_A^p(\mathcal{G}(\rho_2))\|_{\mathrm{tvd}} \\ &\quad\leqslant d_{\mathrm{SORT}} \cdot \left((3200/\beta) \cdot \Delta_{p-1}(n^{\mathrm{rs}} \cdot b) + \Delta_{p-1}(m) + \alpha \cdot m^{-5} \cdot (p/\beta)^{50/\beta}\right) \\ &\quad\leqslant d_{\mathrm{SORT}} \cdot \left((3200/\beta + 1) \cdot \Delta_{p-1}(m) + \alpha \cdot m^{-5} \cdot (p/\beta)^{50/\beta}\right) \qquad (\text{as } n^{\mathrm{rs}} \cdot b > m) \\ &\quad\leqslant \left(\frac{4 \cdot 10^6 \cdot c_{\mathrm{SORT}}}{\beta^2}\right)^{p-1} \cdot m^{-5} \cdot \alpha \cdot \left(\frac{3.2 \cdot 10^5 \cdot c_{\mathrm{SORT}}}{\beta^2}\right) \cdot (((p-1)/\beta)^{50/\beta} + (p/\beta)^{50/\beta}) \\ &\quad\leqslant \left(\frac{4 \cdot 10^6 \cdot c_{\mathrm{SORT}}}{\beta^2}\right)^{p} \cdot m^{-5} \cdot \alpha \cdot (p/\beta)^{50/\beta}, \end{aligned}$$

completing the proof. $\blacksquare$

This concludes our construction of permutation hiding generators and their analysis.

8 A Multi-Pass Streaming Lower Bound for Matchings

We are now ready to present the proof of our main result in Theorem 1, restated below for the convenience of the reader.

Theorem (Restatement of Result 1)**.** *Suppose that for infinitely many choices of $N \geqslant 1$, there exists $(2N)$-vertex bipartite (r,t)-RS graphs with $r = \alpha \cdot N$ and $t = N^\beta$ for some fixed parameters $\alpha, \beta \in (0,1)$; the parameters α, β can depend on N.*

Then, there exists an $\varepsilon_0 = \varepsilon_0(\alpha, \beta)$ such that the following is true. For any $0 < \varepsilon < \varepsilon_0$, any streaming algorithm that uses $o(\varepsilon^2 \cdot n^{1+\beta/2})$ space on n-vertex bipartite graphs and can determine with constant probability whether the input graph has a perfect matching or its maximum matchings have size at most $(1-\varepsilon) \cdot n/2$ requires

$$\Omega\Big(\frac{\log(1/\varepsilon)}{\log(1/\alpha\beta)}\Big)$$

passes over the stream.

The proof is based on a standard reduction, e.g., in [AR20], from reachability to bipartite matching (a similar reduction also appeared in [CKP+21]). We present the reduction for completeness.

Let $m \geqslant 1$ be an even integer and define the following two permutations:

- $\sigma_=$: the identity permutation over $[m/2]$;
- $\sigma_\times$: the "cross identity" permutation that maps $[m/2]$ to $[m/2+1:m]$ and $[m/2+1:m]$ to $[m/2]$ using identity permutations.

Let $\mathcal{G} = \mathcal{G}_{m,n,p,s,\delta}$ be a permutation hiding generator for m and any integer $p \geqslant 1$ with the parameters n, s, δ as in Theorem 3. We are going to use the indistinguishability of graphs sampled from $\mathcal{G}(\sigma_=)$ versus $\mathcal{G}(\sigma_\times)$ to prove Theorem 1. To do so, we need to turn the graphs sampled from this distribution into bipartite graphs wherein size of the matching, even approximately, is a distinguisher for the sample. This is done in the following.

The bipartite graph construction

Let $G = (V, E)$ be sampled from $\mathcal{G}(\sigma_=)$ or $\mathcal{G}(\sigma_\times)$. Create the following bipartite graph H from G:

- The vertices of H are $(L \cup S)$ and $(R \cup T)$ on each side defined as follows:
  - (i) Let L and R be each a separate copy of V. For any vertex $v_i \in V$, we use v_i^{L} and v_i^{R} to denote the copy of v_i in L and R, respectively.
  - (ii) Additionally, create two new sets S and T of vertices with size $m/2$ each.
- The edges of H are E_H plus a matching M defined as follows:
  - (i) For any $i \in [m/2]$ and the i-th vertex $v_i \in \textsc{First}G$ (resp. $v_i \in \textsc{Last}G$), add the edge between i-th vertex $u_i \in S$ and $v_i^{\text{R}} \in R$ (resp. $u_i \in T$ and $v_i^{\text{L}} \in L$) to E_H. Moreover, for any edge $(u_i, v_j) \in E$, add the edge between $u_i^{\text{L}} \in L$ and $v_j^{\text{R}} \in R$.
  - (ii) Finally, for every $v_i \in V$, add the edge between $v_i^{\text{L}} \in L$ and $v_i^{\text{R}} \in R$ to M.

This way, we have a bipartite graph $H = (L \cup S, R \cup T, E_H \cup G)$ associated with G with $n + m/2$ vertices on each side of the bipartition. The following lemma establishes the key property of H.

Lemma 8.1. *Consider a bipartite graph H associated with a graph G defined as above. We have:*

- (i) *if $G \sim \mathcal{G}(\sigma_=)$, then H has a perfect matching of size $n + m/2$;*
- (ii) *on the other hand, if $G \sim \mathcal{G}(\sigma_\times)$, the maximum matching size in H is n.*

Proof. The idea in both cases is to start with the matching M in H and consider augmenting it. We prove each part separately.

Proof of Item (*i*). Consider any vertex $u_i \in S$ in H. These vertices are all unmatched by M. Moreover, u_i only has a single edge to some vertex $v_i^{\mathrm{R}} \in R$. Consider the original copy $v_i \in \textsc{First}G$ of v_i^{R} in the original graph G. Since G is a permutation graph for $\sigma_=$, there is a path $P(v_i)$ in G as follows:

$$P(v_i) := v_i \in \textsc{First}G \;\to\; w_1 \;\to\; w_2 \;\to\; \cdots \;\rightsquigarrow\; \sigma_=(v_i) \in \textsc{Last}G.$$

This path, translates to the following augmenting path in H for the matching M:

$$u_i \in S \;\to_{E_H}\; v_i^{\mathrm{R}} \;\to_M\; v_i^{\mathrm{L}} \;\to_{E_H}\; w_1^{\mathrm{R}} \;\to_M\; w_1^{\mathrm{L}} \;\to_{E_H}\; \cdots \;\rightsquigarrow\; \sigma_=(v_i^{\mathrm{L}}) \;\to_{E_H}\; u_i \in T,$$

using the fact that $\sigma_=$ is the identity matching. Thus, this is a valid augmenting path.

Moreover, for all vertices $u_i \in S$, these augmenting paths should be vertex disjoint. This is because these augmenting paths follow the same paths from $v_i \in \textsc{First}G$ to $v_i \in \textsc{Last}G$ and the paths from the first layer of a permutation graph to its last layer are vertex disjoint (if two paths collide, then starting point of each of these paths, can reach two separate vertices in $\textsc{Last}G$, violating the permutation graph property).

This implies that the matching M of size n admits $m/2$ vertex-disjoint augmenting paths, thus by augmenting it we obtain a matching of size $n + m/2$. Given the bound on the number of vertices of the graph, this is a perfect matching.

Proof of Item (*ii*). All unmatched vertices by M in $L \cup S$ in H actually belong to S. Consider any vertex $u_i \in S$ then and recall it has a single edge to some vertex $v_i^{\mathrm{R}} \in R$. For u_i to be part of an augmenting path, it needs to reach some vertex $w_j \in T$. Such a vertex w_j in turn only has one edge from a single vertex $z_j^{\mathrm{L}} \in L$ by construction.

Using the correspondence between augmenting paths in H and the directed paths in G, we thus need to have a path from $v_i \in \textsc{First}G$ to $z_j \in \textsc{Last}G$. Moreover, by the construction of H, we additionally need both v_i to be in the first $m/2$ vertices of $\textsc{First}G$ and z_j to be in the first $m/2$ vertices of $\textsc{Last}G$. But, since G is a permutation graph for $\sigma_\times$, $v_i \in \textsc{First}G$ can only reach $\sigma_\times(v_i) \in \textsc{Last}G$ which cannot be among the first $m/2$ vertices.

This implies that in this case, the matching M has no augmenting paths in H and is thus a maximum matching of size n. ∎

We now use this lemma to conclude the proof of Theorem 1.

Concluding the proof

Let A be any p-pass s-space streaming algorithm for the maximum matching problem on bipartite graphs with $n + m/2$ vertices on each side of the bipartition that can distinguish between graphs with a perfect matching versus ones with maximum matching of size at most n. We turn A into a p-pass s-space streaming algorithm B for distinguishing between graphs sampled from $\mathcal{G}(\sigma_=)$ and $\mathcal{G}(\sigma_\times)$. We again use the extra power provided in Definition 4.5 to the streaming algorithms.

Given a graph G in the stream, the algorithm B can first create vertices of H and edges in M plus edges from S to R and T to L before even reading the edges of G, and pass these edges of H to A; then, it starts reading the stream of edges of G and each edge (u, v) translates into an edge $(u^{\mathrm{L}}, v^{\mathrm{R}})$ of H by construction, which B passes to A again. This way, B can implement pass of A and at the beginning of the next pass, it again creates edges of M and edges from S to R and T to L in H before reading its own stream, and continues as before until all p passes are finished.

We thus have,

$$\delta \geqslant \|\textsc{mem}_B^p(\mathcal{G}(\sigma_=)) - \textsc{mem}_B^p(\mathcal{G}(\sigma_\times))\|_{\mathrm{tvd}} = \|\textsc{mem}_A^p(H(\mathcal{G}(\sigma_=))) - \textsc{mem}_A^p(H(\mathcal{G}(\sigma_\times)))\|_{\mathrm{tvd}},$$

where δ on the left is the maximum advantage because of the indistinguishability guarantee of $\mathcal{G}$ in Theorem 3.

On the other hand, by Lemma 8.1, size of the maximum matching in $H(\mathcal{G}(\sigma_=)))$ is always $n + m/2$ and in $H(\mathcal{G}(\sigma_\times))$ is always n. This, together with Fact A.8 (on probability of success of distinguishing distributions via a single sample) implies that A also cannot distinguish between these two cases of perfect matching versus maximum matching of size n with probability better than $1/2 + \delta$.

We only now need to instantiate the parameters of A. In the following, let $n_H := n + m/2$ denote the number of vertices on each side of the bipartition of H and $\varepsilon = m/4n$, so that in one case H has a perfect matching of size n_H and in the other case its maximum matching size is at most $(1 - \varepsilon) \cdot n_H$. We further have,

$$n_H = n + m/2 = \Theta(\frac{1}{\alpha \cdot \beta^2})^p \cdot \Theta(m), \qquad \text{(number of vertices)}$$

$$s := o(m^{1+\beta/2}), \qquad \text{(space of streaming algorithm)}$$

by the guarantees of Theorem 3. By calculating parameters p and s in terms of ε and n_H, we get,

$$p = \Omega\Big(\frac{\log(n_H/m)}{\log(1/\alpha \cdot \beta^2)}\Big) = \Omega\Big(\frac{\log(1/\varepsilon)}{\log(1/\alpha\beta)}\Big),$$
$$s = o(m^{1+\beta/2}) = o((\alpha \cdot \beta^2)^{p\cdot(1+\beta/2)} \cdot n_H^{1+\beta/2}) = o(\varepsilon^2 \cdot n_H^{1+\beta/2}).$$

We set our range of ε to be $\varepsilon = \Omega(n^{-\beta/4})$. We just have to see that for this range of ε and p, our advantage remains low. From Theorem 3,

$$\begin{aligned}
\delta &\leqslant (p/\beta)^{\Theta(1/\beta)} \cdot \Theta(1/\beta)^{2p} \cdot m^{-5} \\
&\leqslant \left(\frac{\log(1/\varepsilon)}{\log(1/\alpha\beta)}\right)^{\Theta(1/\beta)} \cdot \Theta(1/\beta)^{\log(1/\varepsilon)/\log(1/\alpha\beta)} \cdot m^{-5} \cdot (1/\beta)^{\Theta(1/\beta)} \\
&\leqslant n \cdot \Theta(1)^{\log(1/\varepsilon)} \cdot (4\varepsilon \cdot n)^{-5} \cdot (1/\beta)^{\Theta(1/\beta)} \\
&\leqslant \Theta(n^{1+\beta/4} \cdot n^{(1-\beta/4)\cdot(-5)}) \leqslant 1/\mathrm{poly}(n).
\end{aligned}$$

This concludes the proof of Theorem 1.

Remark 1. Our lower bound in Theorem 1 can also be extended to some other fundamental problems including **reachability** and **shortest path** studied extensively in the streaming model in [FKM+08, GO13, CGMV20, AR20, CKP+21], because permutation hiding graphs also provide a lower bound for those problems; see [CKP+21].

In particular, under the (plausible) hypothesis that β can be $\Omega(1)$, by using a parameter $m/n \approx n^{-\beta/6}$ in the construction of permutation hiding graphs of Theorem 3 (roughly, setting $\varepsilon \approx n^{-\beta/6}$), we obtain a **lower bound of $\Omega(\log n)$ passes for solving directed reachability or shortest path via semi-streaming algorithms**.

This slightly improves the lower bound of $\Omega(\log n/\log\log n)$ passes for these problems obtained in [GO13, CGMV20, CKP+21] (albeit under the hypothesis that $\beta = \Omega(1)$; using the current state-of-the-art bound of $\Omega(1/\log\log n)$ on β leads to asymptotically same bounds as in [GO13, CGMV20, CKP+21]).

Acknowledgement

We are thankful to Soheil Behnezhad, Surya Teja Gavva, Michael Kapralov, and Huacheng Yu for helpful discussions. We are also grateful to Vladimir V. Podolskii for pointers to the sorting network literature.

References

[A22] S. Assadi. A two-pass (conditional) lower bound for semi-streaming maximum matching. In J. S. Naor and N. Buchbinder, editors, *Proceedings of the 2022 ACM-SIAM Symposium on Discrete Algorithms, SODA 2022, Virtual Conference / Alexandria, VA, USA, January 9 - 12, 2022*, pages 708–742. SIAM, 2022. 1, 2, 3, 4, 5, 10, 11, 13, 14, 20

[A23] S. Assadi. A simple (1-ϵ)-approximation semi-streaming algorithm for maximum (weighted) matching. *CoRR*, abs/2307.02968, 2023. 1

[AB19] S. Assadi and A. Bernstein. Towards a unified theory of sparsification for matching problems. In *2nd Symposium on Simplicity in Algorithms, SOSA@SODA 2019, January 8-9, 2019 - San Diego, CA, USA*, pages 11:1–11:20, 2019. 19

[AB21] S. Assadi and S. Behnezhad. Beating two-thirds for random-order streaming matching. In N. Bansal, E. Merelli, and J. Worrell, editors, *48th International Colloquium on Automata, Languages, and Programming, ICALP 2021, July 12-16, 2021, Glasgow, Scotland (Virtual Conference)*, volume 198 of *LIPIcs*, pages 19:1–19:13. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021. 1, 4, 20

[ABB+19] S. Assadi, M. Bateni, A. Bernstein, V. S. Mirrokni, and C. Stein. Coresets meet EDCS: algorithms for matching and vertex cover on massive graphs. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, pages 1616–1635, 2019. 1

[ABKL23] S. Assadi, S. Behnezhad, S. Khanna, and H. Li. On regularity lemma and barriers in streaming and dynamic matching. In *STOC '23: 55th Annual ACM SIGACT Symposium on Theory of Computing*. ACM, 2023. 1, 3, 20

[Abl93] F. M. Ablayev. Lower bounds for one-way probabilistic communication complexity. In *Automata, Languages and Programming, 20nd International Colloquium, ICALP93, Lund, Sweden, July 5-9, 1993, Proceedings*, pages 241–252, 1993. 10, 11

[ACK19] S. Assadi, Y. Chen, and S. Khanna. Polynomial pass lower bounds for graph streaming algorithms. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, STOC 2019, Phoenix, AZ, USA, June 23-26, 2019*, pages 265–276, 2019. 4

[AG11] K. J. Ahn and S. Guha. Linear programming in the semi-streaming model with application to the maximum matching problem. In *Automata, Languages and Programming - 38th International Colloquium, ICALP 2011, Zurich, Switzerland, July 4-8, 2011, Proceedings, Part II*, pages 526–538, 2011. 1

[AG18] K. J. Ahn and S. Guha. Access to data and number of iterations: Dual primal algorithms for maximum matching under resource constraints. *ACM Trans. Parallel Comput.*, 4(4):17:1–17:40, 2018. 1

[AJJ+22] S. Assadi, A. Jambulapati, Y. Jin, A. Sidford, and K. Tian. Semi-streaming bipartite matching in fewer passes and optimal space. In J. S. Naor and N. Buchbinder, editors, *Proceedings of the 2022 ACM-SIAM Symposium on Discrete Algorithms, SODA 2022, Virtual Conference / Alexandria, VA, USA, January 9 - 12, 2022*, pages 627–669. SIAM, 2022. 1, 3, 4

[AKL17] S. Assadi, S. Khanna, and Y. Li. On estimating maximum matching size in graph streams. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, pages 1723–1742, 2017. 1, 2, 3, 4, 20

[AKLY16] S. Assadi, S. Khanna, Y. Li, and G. Yaroslavtsev. Maximum matchings in dynamic graph streams and the simultaneous communication model. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*, pages 1345–1364, 2016. 1, 20

[AKS83] M. Ajtai, J. Komlós, and E. Szemerédi. An O(n log n) sorting network. In D. S. Johnson, R. Fagin, M. L. Fredman, D. Harel, R. M. Karp, N. A. Lynch, C. H. Papadimitriou, R. L. Rivest, W. L. Ruzzo, and J. I. Seiferas, editors, *Proceedings of the 15th Annual ACM Symposium on Theory of Computing, 25-27 April, 1983, Boston, Massachusetts, USA*, pages 1–9. ACM, 1983. 16, 20, 21

[AKSY20] S. Assadi, G. Kol, R. Saxena, and H. Yu. Multi-pass graph streaming lower bounds for cycle counting, max-cut, matching size, and other problems. In *61st Annual IEEE Symposium on Foundations of Computer Science, FOCS (to appear)*, 2020. 1, 4, 11

[Alo02] N. Alon. Testing subgraphs in large graphs. *Random Struct. Algorithms*, 21(3-4):359–370, 2002. 19

[ALT21] S. Assadi, S. C. Liu, and R. E. Tarjan. An auction algorithm for bipartite matching in streaming and massively parallel computation models. In H. V. Le and V. King, editors, *4th Symposium on Simplicity in Algorithms, SOSA 2021, Virtual Conference, January 11-12, 2021*, pages 165–171. SIAM, 2021. 1, 3

[AMS12] N. Alon, A. Moitra, and B. Sudakov. Nearly complete graphs decomposable into large induced matchings and their applications. In *Proceedings of the 44th Symposium on Theory of Computing Conference, STOC 2012, New York, NY, USA, May 19 - 22, 2012*, pages 1079–1090, 2012. 2, 19

[AN21] S. Assadi and V. N. Graph streaming lower bounds for parameter estimation and property testing via a streaming XOR lemma. In S. Khuller and V. V. Williams, editors, *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*, pages 612–625. ACM, 2021. 1, 4, 10, 22

[AR20] S. Assadi and R. Raz. Near-quadratic lower bounds for two-pass graph streaming algorithms. In *61st IEEE Annual Symposium on Foundations of Computer Science, FOCS 2020, Durham, NC, USA, November 16-19, 2020*, pages 342–353. IEEE, 2020. 1, 2, 3, 4, 14, 17, 20, 59, 61

[AS06] N. Alon and A. Shapira. A characterization of easily testable induced subgraphs. *Combinatorics, Probability & Computing*, 15(6):791–805, 2006. 19

[AS22] S. Assadi and V. Shah. An asymptotically optimal algorithm for maximum matching in dynamic streams. In M. Braverman, editor, *13th Innovations in Theoretical Computer Science Conference, ITCS 2022, January 31 - February 3, 2022, Berkeley, CA, USA*, volume 215 of *LIPIcs*, pages 9:1–9:23. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022. 1

[AS23] S. Assadi and J. Sundaresan. (Noisy) gap cycle counting strikes back: Random order streaming lower bounds for connected components and beyond. In *STOC '23: 55th Annual ACM SIGACT Symposium on Theory of Computing*. ACM, 2023. 1, 4, 5, 10, 11

[Bat68] K. E. Batcher. Sorting networks and their applications. In *American Federation of Information Processing Societies: AFIPS Conference Proceedings: 1968 Spring Joint Computer Conference, Atlantic City, NJ, USA, 30 April - 2 May 1968*, volume 32 of *AFIPS Conference Proceedings*, pages 307–314. Thomson Book Company, Washington D.C., 1968.

[BBCR10] B. Barak, M. Braverman, X. Chen, and A. Rao. How to compress interactive communication. In *Proceedings of the 42nd ACM Symposium on Theory of Computing, STOC 2010, 5-8 June 2010*, pages 67–76, 2010. 12

[BDL21] A. Bernstein, A. Dudeja, and Z. Langley. A framework for dynamic matching in weighted graphs. In S. Khuller and V. V. Williams, editors, *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*, pages 668–681. ACM, 2021. 1, 3

[Ber20] A. Bernstein. Improved bounds for matching in random-order streams. In *47th International Colloquium on Automata, Languages, and Programming, ICALP 2020, July 8-11, 2020, Saarbrücken, Germany (Virtual Conference)*, pages 12:1–12:13, 2020. 1

[BGW20] M. Braverman, S. Garg, and D. P. Woodruff. The coin problem with applications to data streams. In S. Irani, editor, *61st IEEE Annual Symposium on Foundations of Computer Science, FOCS 2020, Durham, NC, USA, November 16-19, 2020*, pages 318–329. IEEE, 2020. 21

[BLM93] Y. Birk, N. Linial, and R. Meshulam. On the uniform-traffic capacity of single-hop interconnections employing shared directional multichannels. *IEEE Transactions on Information Theory*, 39(1):186–191, 1993. 19

[BR11] M. Braverman and A. Rao. Information equals amortized communication. In *IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011, October 22-25, 2011*, pages 748–757, 2011. 12, 13

[BRWY13] M. Braverman, A. Rao, O. Weinstein, and A. Yehudayoff. Direct products in communication complexity. In *54th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2013, 26-29 October, 2013, Berkeley, CA, USA*, pages 746–755, 2013. 12

[BS15] M. Bury and C. Schwiegelshohn. Sublinear estimation of weighted matchings in dynamic data streams. In *Algorithms - ESA 2015 - 23rd Annual European Symposium, September 14-16, 2015, Proceedings*, pages 263–274, 2015. 1

[CCE+16] R. Chitnis, G. Cormode, H. Esfandiari, M. Hajiaghayi, A. McGregor, M. Monemizadeh, and S. Vorotnikova. Kernelization via sampling with applications to finding matchings and related problems in dynamic graph streams. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, January 10-12, 2016*, pages 1326–1344, 2016. 1

[CCHM15] R. H. Chitnis, G. Cormode, M. T. Hajiaghayi, and M. Monemizadeh. Parameterized streaming: Maximal matching and vertex cover. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2015, San Diego, CA, USA, January 4-6, 2015*, pages 1234–1251, 2015. 1

[CDK19] G. Cormode, J. Dark, and C. Konrad. Independent sets in vertex-arrival streams. In *46th International Colloquium on Automata, Languages, and Programming, ICALP 2019, July 9-12, 2019, Patras, Greece*, pages 45:1–45:14, 2019. 20

[CF13] D. Conlonl and J. Fox. Graph removal lemmas. *Surveys in combinatorics 2013*, 409:1, 2013. 2, 19

[CGMV20] A. Chakrabarti, P. Ghosh, A. McGregor, and S. Vorotnikova. Vertex ordering problems in directed graph streams. In *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms, SODA 2020, Salt Lake City, UT, USA, January 5-8, 2020*, pages 1786–1802, 2020. 4, 61

[CGQ15] C. Chekuri, S. Gupta, and K. Quanrud. Streaming algorithms for submodular function maximization. In M. M. Halldórsson, K. Iwama, N. Kobayashi, and B. Speckmann, editors, *Automata, Languages, and Programming - 42nd International Colloquium, ICALP 2015, Kyoto, Japan, July 6-10, 2015, Proceedings, Part I*, volume 9134 of *Lecture Notes in Computer Science*, pages 318–330. Springer, 2015. 1

[Chv92] V. Chvatal. Lecture notes on the new AKS sorting network. Technical report, Rutgers University, 1992. 4, 16, 20, 21

[CJMM17] G. Cormode, H. Jowhari, M. Monemizadeh, and S. Muthukrishnan. The sparse awakens: Streaming algorithms for matching size estimation in sparse graphs. In *25th Annual European Symposium on Algorithms, ESA 2017, September 4-6, 2017*, pages 29:1–29:15, 2017. 1

[CK14] A. Chakrabarti and S. Kale. Submodular maximization meets streaming: Matchings, matroids, and more. In J. Lee and J. Vygen, editors, *Integer Programming and Combinatorial Optimization - 17th International Conference, IPCO 2014, Bonn, Germany, June 23-25, 2014. Proceedings*, volume 8494 of *Lecture Notes in Computer Science*, pages 210–221. Springer, 2014. 1

[CK18] D. Chakrabarty and S. Khanna. Better and simpler error analysis of the sinkhorn-knopp algorithm for matrix scaling. In R. Seidel, editor, *1st Symposium on Simplicity in Algorithms, SOSA 2018, January 7-10, 2018, New Orleans, LA, USA*, volume 61 of *OASIcs*, pages 4:1–4:11. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018. 5, 12, 32, 34, 74

[CKP+21] L. Chen, G. Kol, D. Paramonov, R. R. Saxena, Z. Song, and H. Yu. Almost optimal super-constant-pass streaming lower bounds for reachability. In S. Khuller and V. V.

Williams, editors, *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*, pages 570–583. ACM, 2021. 1, 2, 3, 4, 6, 7, 8, 9, 10, 11, 13, 14, 16, 17, 20, 22, 40, 59, 61

[CS14] M. Crouch and D. S. Stubbs. Improved streaming algorithms for weighted matching, via unweighted matching. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2014, September 4-6, 2014*, pages 96–104, 2014. doi:10.4230/LIPIcs.APPROX-RANDOM.2014.96. 1

[CSWY01] A. Chakrabarti, Y. Shi, A. Wirth, and A. C. Yao. Informational complexity and the direct sum problem for simultaneous message complexity. In *42nd Annual Symposium on Foundations of Computer Science, FOCS 2001, 14-17 October 2001*, pages 270–278, 2001. 12

[CT06] T. M. Cover and J. A. Thomas. *Elements of information theory (2. ed.)*. Wiley, 2006. 71

[DK20] J. Dark and C. Konrad. Optimal lower bounds for matching and vertex cover in dynamic graph streams. In S. Saraf, editor, *35th Computational Complexity Conference, CCC 2020, July 28-31, 2020, Saarbrücken, Germany (Virtual Conference)*, volume 169 of *LIPIcs*, pages 30:1–30:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020. 1

[dW08] R. de Wolf. A brief introduction to fourier analysis on the boolean cube. *Theory Comput.*, 1:1–20, 2008. 12

[EHL+15] H. Esfandiari, M. T. Hajiaghayi, V. Liaghat, M. Monemizadeh, and K. Onak. Streaming algorithms for estimating the matching size in planar graphs and beyond. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2015, January 4-6, 2015*, pages 1217–1233, 2015. 1

[EHM16] H. Esfandiari, M. Hajiaghayi, and M. Monemizadeh. Finding large matchings in semi-streaming. In C. Domeniconi, F. Gullo, F. Bonchi, J. Domingo-Ferrer, R. Baeza-Yates, Z. Zhou, and X. Wu, editors, *IEEE International Conference on Data Mining Workshops, ICDM Workshops 2016, December 12-15, 2016, Barcelona, Spain*, pages 608–614. IEEE Computer Society, 2016. 1

[EKMS12] S. Eggert, L. Kliemann, P. Munstermann, and A. Srivastav. Bipartite matching in the semi-streaming model. *Algorithmica*, 63(1-2):490–508, 2012. 1

[FHM+20] A. Farhadi, M. T. Hajiaghayi, T. Mai, A. Rao, and R. A. Rossi. Approximate maximum matching in random streams. In *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms, SODA 2020, Salt Lake City, UT, USA, January 5-8, 2020*, pages 1773–1785, 2020. 1

[FHS17] J. Fox, H. Huang, and B. Sudakov. On graphs decomposable into induced matchings of linear sizes. *Bulletin of the London Mathematical Society*, 49(1):45–57, 2017. 2, 19

[FKM+05] J. Feigenbaum, S. Kannan, A. McGregor, S. Suri, and J. Zhang. On graph problems in a semi-streaming model. *Theor. Comput. Sci.*, 348(2-3):207–216, 2005. 1, 3

[FKM+08] J. Feigenbaum, S. Kannan, A. McGregor, S. Suri, and J. Zhang. Graph distances in the data-stream model. *SIAM J. Comput.*, 38(5):1709–1727, 2008. 61

[FLN+02] E. Fischer, E. Lehman, I. Newman, S. Raskhodnikova, R. Rubinfeld, and A. Samorodnitsky. Monotonicity testing over general poset domains. In *Proceedings on 34th Annual ACM Symposium on Theory of Computing, May 19-21, 2002, Montréal, Québec, Canada*, pages 474–483, 2002. 2, 3, 5, 19

[FMU22] M. Fischer, S. Mitrovic, and J. Uitto. Deterministic $(1+\epsilon)$-approximate maximum matching with poly$(1/\epsilon)$ passes in the semi-streaming model and beyond. In S. Leonardi and A. Gupta, editors, *STOC '22: 54th Annual ACM SIGACT Symposium on Theory of Computing, Rome, Italy, June 20 - 24, 2022*, pages 248–260. ACM, 2022. 1, 3

[Fox11] J. Fox. A new proof of the graph removal lemma. *Annals of Mathematics*, 174(1):561–579, 2011. 19

[FS22] M. Feldman and A. Szarf. Maximum matching sans maximal matching: A new approach for finding maximum matchings in the data stream model. In A. Chakrabarti and C. Swamy, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2022, September 19-21, 2022, University of Illinois, Urbana-Champaign, USA (Virtual Conference)*, volume 245 of *LIPIcs*, pages 33:1–33:24. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022. 1

[GKK+07] D. Gavinsky, J. Kempe, I. Kerenidis, R. Raz, and R. de Wolf. Exponential separations for one-way quantum communication complexity, with applications to cryptography. *STOC*, pages 516–525, 2007. 4, 5, 10, 11

[GKK12] A. Goel, M. Kapralov, and S. Khanna. On the communication and streaming complexity of maximum bipartite matching. In *Proceedings of the Twenty-third Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '12, pages 468–485. SIAM, 2012. URL http://dl.acm.org/citation.cfm?id=2095116.2095157. 1, 2, 3, 4, 5, 19, 20

[GKMS19] B. Gamlath, S. Kale, S. Mitrovic, and O. Svensson. Weighted matchings via unweighted augmentations. In *Proceedings of the 2019 ACM Symposium on Principles of Distributed Computing, PODC 2019, Toronto, ON, Canada, July 29 - August 2, 2019*, pages 491–500, 2019. 1, 3

[GM08] S. Guha and A. McGregor. Tight lower bounds for multi-pass stream computation via pass elimination. In *Automata, Languages and Programming, 35th International Colloquium, ICALP 2008, July 7-11, 2008, Proceedings, Part I: Tack A: Algorithms, Automata, Complexity, and Games*, pages 760–772, 2008. 21

[GO13] V. Guruswami and K. Onak. Superlinear lower bounds for multipass graph processing. In *Proceedings of the 28th Conference on Computational Complexity, CCC 2013, K.lo Alto, California, USA, 5-7 June, 2013*, pages 287–298, 2013. 1, 2, 3, 4, 61

[Gow01] W. Gowers. Some unsolved problems in additive/combinatorial number theory. *preprint*, 2001. 2, 19

[GT19] V. Guruswami and R. Tao. Streaming hardness of unique games. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2019, September 20-22, 2019, Massachusetts Institute of Technology, Cambridge, MA, USA*, pages 5:1–5:12, 2019. 7

[HGG09] J. Huang, C. Guestrin, and L. Guibas. Fourier theoretic probabilistic inference over permutations. *J. Mach. Learn. Res.*, 10:997–1070, jun 2009. 74, 75

[JRS03] R. Jain, J. Radhakrishnan, and P. Sen. A direct sum theorem in communication complexity via message compression. In *Automata, Languages and Programming, 30th International Colloquium, ICALP 2003, June 30 - July 4, 2003. Proceedings*, pages 300–315, 2003. 12

[Kap13] M. Kapralov. Better bounds for matchings in the streaming model. In *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2013, New Orleans, Louisiana, USA, January 6-8, 2013*, pages 1679–1697, 2013. doi:10.1137/1.9781611973105.121. 1, 2, 3, 20

[Kap21] M. Kapralov. Space lower bounds for approximating maximum matching in the edge arrival model. In D. Marx, editor, *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA 2021, Virtual Conference, January 10 - 13, 2021*, pages 1874–1893. SIAM, 2021. 1, 2, 3, 20

[KKL88] J. Kahn, G. Kalai, and N. Linial. The influence of variables on boolean functions (extended abstract). In *29th Annual Symposium on Foundations of Computer Science, White Plains, New York, USA, 24-26 October 1988*, pages 68–80. IEEE Computer Society, 1988. 5, 12

[KKS14] M. Kapralov, S. Khanna, and M. Sudan. Approximating matching size from random streams. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, Portland, Oregon, USA, January 5-7, 2014*, pages 734–751, 2014. 1

[KKS15] M. Kapralov, S. Khanna, and M. Sudan. Streaming lower bounds for approximating MAX-CUT. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2015, San Diego, CA, USA, January 4-6, 2015*, pages 1263–1282, 2015. 5, 10, 11

[KKTY21] M. Kapralov, R. Krauthgamer, J. Tardos, and Y. Yoshida. Towards tight bounds for spectral sparsification of hypergraphs. In S. Khuller and V. V. Williams, editors, *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*, pages 598–611. ACM, 2021. 19

[KMM12] C. Konrad, F. Magniez, and C. Mathieu. Maximum matching in semi-streaming with few passes. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques - 15th International Workshop, APPROX 2012, and 16th International Workshop, RANDOM 2012, Cambridge, MA, USA, August 15-17, 2012. Proceedings*, pages 231–242, 2012. 1

[KMNT20] M. Kapralov, S. Mitrovic, A. Norouzi-Fard, and J. Tardos. Space efficient approximation to maximum matching size from uniform edge samples. In *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms, SODA 2020, Salt Lake City, UT, USA, January 5-8, 2020*, pages 1753–1772, 2020. 1

[KMT+22] M. Kapralov, A. Musipatla, J. Tardos, D. P. Woodruff, and S. Zhou. Noisy boolean hidden matching with applications. In M. Braverman, editor, *13th Innovations in*

Theoretical Computer Science Conference, ITCS 2022, January 31 - February 3, 2022, Berkeley, CA, USA, volume 215 of *LIPIcs*, pages 91:1–91:19. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022. 4, 5, 10, 11

[KN21] C. Konrad and K. K. Naidu. On two-pass streaming algorithms for maximum bipartite matching. In M. Wootters and L. Sanità, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2021, August 16-18, 2021, University of Washington, Seattle, Washington, USA (Virtual Conference)*, volume 207 of *LIPIcs*, pages 19:1–19:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021. 1, 2, 3, 20

[KNR95] I. Kremer, N. Nisan, and D. Ron. On randomized one-round communication complexity. In *Proceedings of the Twenty-Seventh Annual ACM Symposium on Theory of Computing, 29 May-1 June 1995, Las Vegas, Nevada, USA*, pages 596–605, 1995. 10, 11

[KNS23] C. Konrad, K. K. Naidu, and A. Steward. Maximum matching via maximal matching queries. In P. Berenbrink, P. Bouyer, A. Dawar, and M. M. Kanté, editors, *40th International Symposium on Theoretical Aspects of Computer Science, STACS 2023, March 7-9, 2023, Hamburg, Germany*, volume 254 of *LIPIcs*, pages 41:1–41:22. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2023. 1

[Knu97] D. E. Knuth. *The art of computer programming*, volume 3. Pearson Education, 1997. 20, 78

[Kon15] C. Konrad. Maximum matching in turnstile streams. In *Algorithms - ESA 2015 - 23rd Annual European Symposium, September 14-16, 2015, Proceedings*, pages 840–852, 2015. 1, 20

[Kon18] C. Konrad. A simple augmentation method for matchings with applications to streaming algorithms. In *43rd International Symposium on Mathematical Foundations of Computer Science, MFCS 2018, August 27-31, 2018, Liverpool, UK*, pages 74:1–74:16, 2018. 1

[KT17] S. Kale and S. Tirodkar. Maximum matching in two, three, and a few more passes over graph streams. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2017, August 16-18, 2017, Berkeley, CA, USA*, pages 15:1–15:21, 2017. 1

[LNW14] Y. Li, H. L. Nguyen, and D. P. Woodruff. Turnstile streaming algorithms might as well be linear sketches. In *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 174–183, 2014. 21

[LSZ20] S. C. Liu, Z. Song, and H. Zhang. Breaking the n-pass barrier: A streaming algorithm for maximum weight bipartite matching. *CoRR*, abs/2009.06106, 2020. 1

[LW21] R. Levin and D. Wajc. Streaming submodular matching meets the primal-dual method. In D. Marx, editor, *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA 2021, Virtual Conference, January 10 - 13, 2021*, pages 1914–1933. SIAM, 2021. 1

[McG05] A. McGregor. Finding graph matchings in data streams. In *Approximation, Randomization and Combinatorial Optimization, Algorithms and Techniques, 8th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems, APPROX 2005 and 9th InternationalWorkshop on Randomization and Computation, RANDOM 2005, Berkeley, CA, USA, August 22-24, 2005, Proceedings*, pages 170–181, 2005. 1, 3

[MV16] A. McGregor and S. Vorotnikova. Planar matching in streams revisited. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2016, September 7-9, 2016*, pages 17:1–17:12, 2016. 1

[MV18] A. McGregor and S. Vorotnikova. A simple, space-efficient, streaming algorithm for matchings in low arboricity graphs. In *1st Symposium on Simplicity in Algorithms, SOSA 2018, January 7-10, 2018*, pages 14:1–14:4, 2018. 1

[PP89] B. Parker and I. Parberry. Constructing sorting networks from k-sorters. *Inf. Process. Lett.*, 33(3):157–162, 1989. 4, 16, 20, 77, 78

[PS17] A. Paz and G. Schwartzman. A $(2+\varepsilon)$-approximation for maximum weight matching in the semi-streaming model. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, pages 2153–2161, 2017. 1

[RS78] I. Z. Ruzsa and E. Szemerédi. Triple systems with no six points carrying three triangles. *Combinatorics (Keszthely, 1976), Coll. Math. Soc. J. Bolyai*, 18:939–945, 1978. 2, 19

[Tir18] S. Tirodkar. Deterministic algorithms for maximum matching on general graphs in the semi-streaming model. In *38th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2018, December 11-13, 2018, Ahmedabad, India*, pages 39:1–39:16, 2018. 1, 3

[TV06] T. Tao and V. H. Vu. *Additive combinatorics*, volume 105. Cambridge University Press, 2006. 19

[VY11] E. Verbin and W. Yu. The streaming complexity of cycle counting, sorting by reversals, and other problems. In *Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2011, January 23-25, 2011*, pages 11–25, 2011. 4, 5, 6, 10, 11

[Yao82] A. C. Yao. Theory and applications of trapdoor functions (extended abstract). In *23rd Annual Symposium on Foundations of Computer Science, Chicago, Illinois, USA, 3-5 November 1982*, pages 80–91, 1982. 10

[Yu22] H. Yu. Strong XOR lemma for communication with bounded rounds : (extended abstract). In *63rd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2022, Denver, CO, USA, October 31 - November 3, 2022*, pages 1186–1192. IEEE, 2022. 10

A Background on Information Theory

We now briefly introduce some definitions and facts from information theory that are used in our proofs. We refer the interested reader to the text by Cover and Thomas [CT06] for an excellent introduction to this field, and the proofs of the statements used in this Appendix.

For a random variable A, we use $\mathrm{supp}(\mathsf{A})$ to denote the support of A and $\mathrm{dist}(\mathsf{A})$ to denote its distribution. When it is clear from the context, we may abuse the notation and use A directly instead of $\mathrm{dist}(\mathsf{A})$, for example, write $A \sim \mathsf{A}$ to mean $A \sim \mathrm{dist}(\mathsf{A})$, i.e., A is sampled from the distribution of random variable A.

- We denote the *Shannon Entropy* of a random variable A by $\mathbb{H}(\mathsf{A})$, which is defined as:

$$\mathbb{H}(\mathsf{A}) := \sum_{A \in \mathrm{supp}(\mathsf{A})} \Pr(\mathsf{A} = A) \cdot \log(1/\Pr(\mathsf{A} = A)) \tag{25}$$

- The *conditional entropy* of A conditioned on B is denoted by $\mathbb{H}(\mathsf{A} \mid \mathsf{B})$ and defined as:

$$\mathbb{H}(\mathsf{A} \mid \mathsf{B}) := \mathop{\mathbb{E}}_{B \sim \mathsf{B}} [\mathbb{H}(\mathsf{A} \mid \mathsf{B} = B)], \tag{26}$$

  where $\mathbb{H}(\mathsf{A} \mid \mathsf{B} = B)$ is defined in a standard way by using the distribution of A conditioned on the event $\mathsf{B} = B$ in Eq (25).

- The *mutual information* of two random variables A and B is denoted by $\mathbb{I}(\mathsf{A}\,;\mathsf{B})$ and is defined:

$$\mathbb{I}(\mathsf{A}\,;\mathsf{B}) := \mathbb{H}(A) - \mathbb{H}(A \mid B) = \mathbb{H}(B) - \mathbb{H}(B \mid A). \tag{27}$$

- The *conditional mutual information* $\mathbb{I}(\mathsf{A}\,;\mathsf{B} \mid \mathsf{C})$ is $\mathbb{H}(\mathsf{A} \mid \mathsf{C}) - \mathbb{H}(\mathsf{A} \mid \mathsf{B}, \mathsf{C})$ and hence by linearity of expectation:

$$\mathbb{I}(\mathsf{A}\,;\mathsf{B} \mid \mathsf{C}) = \mathop{\mathbb{E}}_{C \sim \mathsf{C}} [\mathbb{I}(\mathsf{A}\,;\mathsf{B} \mid \mathsf{C} = C)]. \tag{28}$$

A.1 Useful Properties of Entropy and Mutual Information

We shall use the following basic properties of entropy and mutual information throughout.

Fact A.1. *Let* A, B, C, *and* D *be four (possibly correlated) random variables.*

1. $0 \leqslant \mathbb{H}(\mathsf{A}) \leqslant \log|\mathrm{supp}(\mathsf{A})|$. *The right equality holds iff* $\mathrm{dist}(\mathsf{A})$ *is uniform.*
2. $\mathbb{I}(\mathsf{A}\,;\mathsf{B} \mid \mathsf{C}) \geqslant 0$. *The equality holds iff* A *and* B *are* independent *conditioned on* C.
3. Conditioning on a random variable reduces entropy*:* $\mathbb{H}(\mathsf{A} \mid \mathsf{B}, \mathsf{C}) \leqslant \mathbb{H}(\mathsf{A} \mid \mathsf{B})$. *The equality holds iff* $\mathsf{A} \perp \mathsf{C} \mid \mathsf{B}$.
4. Subadditivity of entropy*:* $\mathbb{H}(\mathsf{A}, \mathsf{B} \mid \mathsf{C}) \leqslant \mathbb{H}(\mathsf{A} \mid C) + \mathbb{H}(\mathsf{B} \mid \mathsf{C})$.
5. Chain rule for entropy*:* $\mathbb{H}(\mathsf{A}, \mathsf{B} \mid \mathsf{C}) = \mathbb{H}(\mathsf{A} \mid \mathsf{C}) + \mathbb{H}(\mathsf{B} \mid \mathsf{C}, \mathsf{A})$.
6. Chain rule for mutual information*:* $\mathbb{I}(\mathsf{A}, \mathsf{B}\,;\mathsf{C} \mid \mathsf{D}) = \mathbb{I}(\mathsf{A}\,;\mathsf{C} \mid \mathsf{D}) + \mathbb{I}(\mathsf{B}\,;\mathsf{C} \mid \mathsf{A}, \mathsf{D})$.
7. Data processing inequality*: for a function* $f(\mathsf{A})$ *of* A, $\mathbb{I}(f(\mathsf{A})\,;\mathsf{B} \mid \mathsf{C}) \leqslant \mathbb{I}(\mathsf{A}\,;\mathsf{B} \mid \mathsf{C})$.

We also use the following two standard propositions, regarding the effect of conditioning on mutual information.

Proposition A.2. *For random variables* $\mathsf{A}, \mathsf{B}, \mathsf{C}, \mathsf{D}$*, if* $\mathsf{A} \perp \mathsf{D} \mid \mathsf{C}$*, then,*

$$\mathbb{I}(\mathsf{A}\,;\mathsf{B} \mid \mathsf{C}) \leqslant \mathbb{I}(\mathsf{A}\,;\mathsf{B} \mid \mathsf{C}, \mathsf{D}).$$

Proof. Since A and D are independent conditioned on C, by Fact A.1-(3), $\mathbb{H}(\mathsf{A} \mid \mathsf{C}) = \mathbb{H}(\mathsf{A} \mid \mathsf{C}, \mathsf{D})$ and $\mathbb{H}(\mathsf{A} \mid \mathsf{C}, \mathsf{B}) \geqslant \mathbb{H}(\mathsf{A} \mid \mathsf{C}, \mathsf{B}, \mathsf{D})$. We have,

$$\begin{aligned}\mathbb{I}(\mathsf{A}\,;\mathsf{B} \mid \mathsf{C}) &= \mathbb{H}(\mathsf{A} \mid \mathsf{C}) - \mathbb{H}(\mathsf{A} \mid \mathsf{C}, \mathsf{B}) = \mathbb{H}(\mathsf{A} \mid \mathsf{C}, \mathsf{D}) - \mathbb{H}(\mathsf{A} \mid \mathsf{C}, \mathsf{B}) \\ &\leqslant \mathbb{H}(\mathsf{A} \mid \mathsf{C}, \mathsf{D}) - \mathbb{H}(\mathsf{A} \mid \mathsf{C}, \mathsf{B}, \mathsf{D}) = \mathbb{I}(\mathsf{A}\,;\mathsf{B} \mid \mathsf{C}, \mathsf{D}). \qquad \blacksquare\end{aligned}$$

Proposition A.3. *For random variables* $\mathsf{A}, \mathsf{B}, \mathsf{C}, \mathsf{D}$*, if* $\mathsf{A} \perp \mathsf{D} \mid \mathsf{B}, \mathsf{C}$*, then,*

$$\mathbb{I}(\mathsf{A}\,;\mathsf{B} \mid \mathsf{C}) \geqslant \mathbb{I}(\mathsf{A}\,;\mathsf{B} \mid \mathsf{C}, \mathsf{D}).$$

Proof. Since $\mathsf{A} \perp \mathsf{D} \mid \mathsf{B}, \mathsf{C}$, by Fact A.1-(3), $\mathbb{H}(\mathsf{A} \mid \mathsf{B}, \mathsf{C}) = \mathbb{H}(\mathsf{A} \mid \mathsf{B}, \mathsf{C}, \mathsf{D})$. Moreover, since conditioning can only reduce the entropy (again by Fact A.1-(3)),

$$\begin{aligned}\mathbb{I}(\mathsf{A}\,;\mathsf{B} \mid \mathsf{C}) &= \mathbb{H}(\mathsf{A} \mid \mathsf{C}) - \mathbb{H}(\mathsf{A} \mid \mathsf{B}, \mathsf{C}) \geqslant \mathbb{H}(\mathsf{A} \mid \mathsf{D}, \mathsf{C}) - \mathbb{H}(\mathsf{A} \mid \mathsf{B}, \mathsf{C}) \\ &= \mathbb{H}(\mathsf{A} \mid \mathsf{D}, \mathsf{C}) - \mathbb{H}(\mathsf{A} \mid \mathsf{B}, \mathsf{C}, \mathsf{D}) = \mathbb{I}(\mathsf{A}\,;\mathsf{B} \mid \mathsf{C}, \mathsf{D}). \qquad \blacksquare\end{aligned}$$

A.2 Measures of Distance Between Distributions

We use two main measures of distance (or divergence) between distributions, namely the *Kullback-Leibler divergence* (KL-divergence) and the *total variation distance*.

KL-divergence. For two distributions μ and ν over the same probability space, the **Kullback-Leibler (KL) divergence** between μ and ν is denoted by $\mathbb{D}(\mu \mid\mid \nu)$ and defined as:

$$\mathbb{D}(\mu \mid\mid \nu) := \mathop{\mathbb{E}}_{a \sim \mu}\left[\log \frac{\mu(a)}{\nu(a)}\right]. \tag{29}$$

We also have the following relation between mutual information and KL-divergence.

Fact A.4. *For random variables* $\mathsf{A}, \mathsf{B}, \mathsf{C}$*,*

$$\mathbb{I}(\mathsf{A}\,;\mathsf{B} \mid \mathsf{C}) = \mathop{\mathbb{E}}_{(B,C) \sim (\mathsf{B},\mathsf{C})}\Big[\mathbb{D}(\mathrm{dist}(\mathsf{A} \mid \mathsf{B} = B, \mathsf{C} = C) \mid\mid \mathrm{dist}(\mathsf{A} \mid \mathsf{C} = C))\Big].$$

We use the following standard facts about KL-divergence.

Fact A.5 (Chain rule of KL-divergence)**.** *Let* $\mu(\mathsf{X}, \mathsf{Y})$ *and* $\nu(\mathsf{X}, \mathsf{Y})$ *be two distributions for random variables* X, Y*. Then,*

$$\mathbb{D}(\mu(\mathsf{X}, \mathsf{Y}) \mid\mid \nu(\mathsf{X}, \mathsf{Y})) = \mathbb{D}(\mu(\mathsf{X}) \mid\mid \nu(\mathsf{X})) + \mathop{\mathbb{E}}_{x \sim \mu(\mathsf{X})} \mathbb{D}(\mu(\mathsf{Y} \mid \mathsf{X} = x) \mid\mid \nu(\mathsf{Y} \mid \mathsf{X} = x)).$$

Moreover, if $\mathsf{X} \perp \mathsf{Y}$ *in* ν *(the second argument of the KL-divergence), then,*

$$\mathbb{D}(\mu(\mathsf{X}, \mathsf{Y}) \mid\mid \nu(\mathsf{X}, \mathsf{Y})) \geqslant \mathbb{D}(\mu(\mathsf{X}) \mid\mid \nu(\mathsf{X})) + \mathbb{D}(\mu(\mathsf{Y}) \mid\mid \nu(\mathsf{Y})).$$

Fact A.6 (Conditioning in KL-divergence)**.** *For any random variable* X *and any event* $\mathcal{E}$*,*

$$\mathbb{D}(\mathsf{X} \mid \mathcal{E} \mid\mid \mathsf{X}) \leqslant \log\left(\frac{1}{\Pr(\mathcal{E})}\right).$$

Moreover, if $\mathcal{E}$ *is a deterministic function of* X*, then this equation holds with equality.*

Total variation distance. We denote the **total variation distance** between two distributions μ and ν on the same support Ω by $\|\mu - \nu\|_{\mathrm{tvd}}$, defined as:

$$\|\mu - \nu\|_{\mathrm{tvd}} := \max_{\Omega' \subseteq \Omega} \left(\mu(\Omega') - \nu(\Omega')\right) = \frac{1}{2} \cdot \sum_{x \in \Omega} |\mu(x) - \nu(x)| \,. \tag{30}$$

We use the following basic properties of total variation distance.

Fact A.7. *Suppose μ and ν are two distributions for $\mathcal{E}$, then, $\mu(\mathcal{E}) \leqslant \nu(\mathcal{E}) + \|\mu - \nu\|_{\mathrm{tvd}}$.*

Fact A.8. *Suppose μ and ν are two distributions with same support Ω; then, given a single sample from either μ or ν, the best probability of successfully deciding whether s came from μ or ν (achieved by the maximum likelihood estimator) is*

$$\frac{1}{2} + \frac{1}{2} \cdot \|\mu - \nu\|_{\mathrm{tvd}}.$$

We also have the following (chain-rule) bound on the total variation distance of joint variables.

Fact A.9. *For any distributions μ and ν on n-tuples $(X_1, \ldots, X_n)$,*

$$\|\mu - \nu\|_{\mathrm{tvd}} \leqslant \sum_{i=1}^{n} \mathop{\mathbb{E}}_{X_{<i} \sim \mu} \|\mu(X_i \mid X_{<i}) - \nu(X_i \mid X_{<i})\|_{\mathrm{tvd}}.$$

A simple consequence of this fact gives us the following "over conditioning" property as well.

Fact A.10. *For any random variables $\mathsf{X}, \mathsf{Y}, \mathsf{Z}$,*

$$\|\mathsf{X} - \mathsf{Y}\|_{\mathrm{tvd}} \leqslant \|\mathsf{XZ} - \mathsf{YZ}\|_{\mathrm{tvd}} = \mathop{\mathbb{E}}_{Z} \|(\mathsf{X} \mid \mathsf{Z} = Z) - (\mathsf{Y} \mid \mathsf{Z} = Z)\|_{\mathrm{tvd}}.$$

Proof. By the non-negativity of TVD and Fact A.9,

$$\|\mathsf{X} - \mathsf{Y}\|_{\mathrm{tvd}} \leqslant \|\mathsf{X} - \mathsf{Y}\|_{\mathrm{tvd}} + \mathop{\mathbb{E}}_{X} \|(\mathsf{Z} \mid \mathsf{X} = X) - (\mathsf{Z} \mid \mathsf{Y} = X)\|_{\mathrm{tvd}} = \|\mathsf{XZ} - \mathsf{YZ}\|_{\mathrm{tvd}}$$

Applying Fact A.9 again gives us

$$\|\mathsf{XZ} - \mathsf{YZ}\|_{\mathrm{tvd}} = \|\mathsf{Z} - \mathsf{Z}\|_{\mathrm{tvd}} + \mathop{\mathbb{E}}_{Z} \|(\mathsf{X} \mid \mathsf{Z} = Z) - (\mathsf{Y} \mid \mathsf{Z} = Z)\|_{\mathrm{tvd}} = \mathop{\mathbb{E}}_{Z} \|(\mathsf{X} \mid \mathsf{Z} = Z) - (\mathsf{Y} \mid \mathsf{Z} = Z)\|_{\mathrm{tvd}},$$

which concludes the proof. ∎

Similarly, we have the following data processing inequality for total variation distance as a consequence of the above.

Fact A.11. *Suppose X and Y are two random variables with the same support Ω and $f : \Omega \to \Omega$ is a fixed function. Then,*

$$\|f(\mathsf{X}) - f(\mathsf{Y})\|_{\mathrm{tvd}} \leqslant \|\mathsf{X} - \mathsf{Y}\|_{\mathrm{tvd}}.$$

Connections Between KL-Divergence and Total Variation Distance

The following Pinsker's inequality bounds the total variation distance between two distributions based on their KL-divergence,

Fact A.12 (Pinsker's inequality)**.** *For any distributions μ and ν, $\|\mu - \nu\|_{\text{tvd}} \leqslant \sqrt{\frac{1}{2} \cdot \mathbb{D}(\mu \mid\mid \nu)}$.*

We shall also use the following strengthening of Pinsker's inequality due to [CK18] that allows to lower bound KL-divergence between two distributions by a combination of ℓ_1- and ℓ_2-distance of the two distributions (instead of purely ℓ_1-distance in the original Pinsker's inequality).

Proposition A.13 (Strengthened Pinsker's Inequality [CK18, KL vs ℓ_1/ℓ_2-inequality])**.** *Given any pair of distributions μ and ν over the same finite domain Ω, define*

$$A := \{x \in \Omega \mid \mu(x) > 2 \cdot \nu(x)\} \quad \text{and} \quad B := \Omega \setminus A.$$

Then,

$$\mathbb{D}(\mu \mid\mid \nu) \geqslant (1 - \ln 2) \cdot \left(\sum_{x \in A} |\mu(x) - \nu(x)| + \sum_{x \in B} \frac{(\mu(x) - \nu(x))^2}{\mu(x)} \right).$$

B Background on Fourier Analysis on Permutations

We review basics of representation theory and Fourier transform on permutations. We refer the interested reader to [HGG09] for more details and background.

Representation theory. We can define representations for any (symmetric) group, including the group of permutations on a fixed domain.

Definition B.1. A **representation** of a group G is a map ρ from G to a set of invertible $d_\rho \times d_\rho$ (complex) matrix operators ($\rho : G \to \mathbb{C}^{d_\rho \times d_\rho}$) which preserves algebraic structure in the sense that for all $\sigma_1, \sigma_2 \in G, \rho(\sigma_1 \circ \sigma_2) = \rho(\sigma_1) \cdot \rho(\sigma_2)$. The matrices which lie in the image of ρ are called the **representation matrices**, and we will refer to d_ρ as the **degree** of the representation.

Two representations ρ_1, ρ_2 are said to be **equivalent** if there exists an invertible matrix C such that for all $\sigma \in G$,

$$\rho_2(\sigma) = C^{-1} \cdot \rho_1(\sigma) \cdot C.$$

Given two representations ρ_1, ρ_2, we write the direct sum of ρ_1 and ρ_2, denoted by $\rho_1 \oplus \rho_2$ as,

$$\rho_1 \oplus \rho_2(\sigma) = \begin{bmatrix} \rho_1(\sigma) & 0 \\ 0 & \rho_2(\sigma) \end{bmatrix}.$$

The degree of $\rho_1 \oplus \rho_2$ is $d_{\rho_1} + d_{\rho_2}$.

Definition B.2. A representation ρ is said to be **reducible** if it can be decomposed as $\rho = \rho_1 \oplus \rho_2$. The set of **irreducible representations** of any group G is the collection of representations (up to equivalence) which are not reducible.

An example of an irreducible representation is the trivial representation, denoted by ρ_0, that takes every group element to $1 \in \mathbb{R}$.

Fourier transform over permutations. Fix any integer $b \geqslant 1$ and let S_b be the set of permutations on $[b]$ in the following. Let $\textsc{RepBasis} := \textsc{RepBasis}(b)$ denote the (finite) set of all irreducible representations of the symmetric group S_b.

For any function $f : S_b \to \mathbb{R}$ and representation $\rho \in \textsc{RepBasis}$, the Fourier transform of f at ρ is a $d_\rho \times d_\rho$ dimensional matrix defined as,

$$\widehat{\rho(f)} := \sum_{\sigma \in S_b} f(\sigma) \cdot \rho(\sigma). \tag{31}$$

The set of Fourier transforms at all representations in $\textsc{RepBasis}$ form the Fourier transform of f.

The Fourier Inversion theorem then implies that for any function $f : S_b \to \mathbb{R}$,

$$f(\sigma) = \frac{1}{b!} \cdot \sum_{\rho \in \textsc{RepBasis}} d_\rho \cdot \textsc{Trace}(\widehat{\rho(f)}^\top \cdot \rho(\sigma)). \tag{32}$$

Fourier Transforms of Distributions Over Permutations

In the following, let $\mathcal{U} := \mathcal{U}_{S_b}$ denote the uniform distribution over S_b and ν be any arbitrary distribution on S_b, where for $\sigma \in S_b$, $\nu(\sigma)$ denote the probability of σ under ν. Moreover, ρ_0 is the trivial representation in $\textsc{RepBasis}$ that maps all permutations to 1×1 dimensional matrix with entry 1. The proofs of all following standard results can be found in [HGG09].

Fact B.3 (c.f. [HGG09, Section 4.2])**.** *For the Fourier transform over permutations,*

1. *For any probability distribution ν over S_b, $\widehat{\rho_0(\nu)} = 1$.*
2. *For the uniform distribution $\mathcal{U}$, $\widehat{\rho_0(\mathcal{U})} = 1$, and for any $\rho \in \textsc{RepBasis} \setminus \{\rho_0\}$, $\widehat{\rho(\mathcal{U})} = \mathbf{0}$, where $\mathbf{0}$ is the $d_\rho \times d_\rho$ dimensional matrix of all zeros.*

We can also use the Fourier convolution theorem to relate Fourier coefficient of distribution on concatenated permutations to each other. Given two distributions ν_1 and ν_2 over S_b, define $\nu := \nu_1 \circ \nu_2$ as the distribution obtained by sampling $\sigma_1 \sim \nu_1$ and $\sigma_2 \sim \nu_2$ independently and returning $\sigma_1 \circ \sigma_2$ as the sample of ν.

Fact B.4 (c.f. [HGG09, Definition 7 and Proposition 8])**.** *For any distributions ν_1 and ν_2 over S_b and $\nu = \nu_1 \circ \nu_2$, and any $\rho \in \textsc{RepBasis}$,*

$$\widehat{\rho(\nu)} = \widehat{\rho(\nu_1)} \cdot \widehat{\rho(\nu_2)}.$$

Finally, we also have the following Plancherel's identity for this Fourier transform.

Proposition B.5 ([HGG09, Proposition 13])**.** *For any distributions ν_1 and ν_2 over S_b,*

$$\sum_{\sigma \in S_b} (\nu_1(\sigma) - \nu_2(\sigma))^2 = \frac{1}{b!} \cdot \sum_{\rho \in \textsc{RepBasis}} d_\rho \cdot \sum_{i,j \in [d_\rho]} \left(\widehat{\rho(\nu_1)} - \widehat{\rho(\nu_2)}\right)_{i,j}^2.$$

C Tightness of Lemma 5.11

Our proof in Lemma 5.11 can be restated as follows. Let $\nu_1, \ldots, \nu_g$ be g distributions over S_b and define $\nu = \nu_1 \circ \cdots \circ \nu_g$ (as defined in Appendix B). Suppose

$$\|\nu_i - \mathcal{U}_{S_b}\|_2^2 = \frac{1}{b!} \cdot \varepsilon_i,$$

for all $i \in [g]$. Then,

$$\|\nu - \mathcal{U}_{S_b}\|_2^2 \leqslant \frac{1}{b!} \cdot \prod_{i=1}^{g} \varepsilon_i.$$

(the proof is verbatim as in Lemma 5.11). We now argue this bound is actually sharp.

Let ν_i for $i \in [g]$ be the following distribution.

$$\nu_i(\sigma) = \begin{cases} \dfrac{1+\sqrt{\varepsilon_i}}{b!} & \text{if } \sigma \text{ has an even number of inversions,} \\ \dfrac{1-\sqrt{\varepsilon_i}}{b!} & \text{otherwise.} \end{cases}$$

The ℓ_2 distance of ν_i from $\mathcal{U}_{S_b}$ is $\varepsilon_i/b!$ by construction. Now, consider the distribution $\nu_1 \circ \nu_2$.

- To get an even permutation, either two even permutations $\sigma_1 \sim \nu_1$ and $\sigma_2 \sim \nu_2$ can be picked, or two odd permutations can be picked. Thus for an even $\sigma \in S_b$,

$$\begin{aligned} \nu_1 \circ \nu_2(\sigma) &= \frac{b!}{2} \cdot \left(\left(\frac{1}{b!}\right)^2 \cdot (1+\sqrt{\varepsilon_1})(1+\sqrt{\varepsilon_2}) \right) + \frac{b!}{2} \cdot \left(\left(\frac{1}{b!}\right)^2 \cdot (1-\sqrt{\varepsilon_1})(1-\sqrt{\varepsilon_2}) \right) \\ &= \frac{1+\sqrt{\varepsilon_1\varepsilon_2}}{b!}. \end{aligned}$$

- To get an odd permutation, we sample either an odd permutation from ν_1 and even from ν_2 or vice-versa. Hence for any odd $\sigma \in S_b$,

$$\begin{aligned} \nu_1 \circ \nu_2(\sigma) &= \frac{b!}{2} \cdot \left(\left(\frac{1}{b!}\right)^2 \cdot (1+\sqrt{\varepsilon_1})(1-\sqrt{\varepsilon_2}) \right) + \frac{b!}{2} \cdot \left(\left(\frac{1}{b!}\right)^2 \cdot (1-\sqrt{\varepsilon_1})(1+\sqrt{\varepsilon_2}) \right) \\ &= \frac{1-\sqrt{\varepsilon_1\varepsilon_2}}{b!}. \end{aligned}$$

If we proceed by induction we see that distribution $\nu = \nu_1 \circ \cdots \nu_g$ is as follows:

$$\nu(\sigma) = \begin{cases} \frac{1}{b!} \cdot \left(1 + \left(\prod_{i=1}^{g} \varepsilon_i\right)^{1/2}\right) & \text{if } \sigma \text{ has even number of inversions and} \\ \frac{1}{b!} \cdot \left(1 - \left(\prod_{i=1}^{g} \varepsilon_i\right)^{1/2}\right) & \text{otherwise.} \end{cases}$$

One can then calculate that the ℓ_2-distance of ν from $\mathcal{U}_{S_b}$ is

$$\frac{1}{b!} \cdot \prod_{i=1}^{\ell} \varepsilon_i,$$

exactly matching the bounds in Lemma 5.11.

D Sorting Networks with Large Comparators

We provide a weaker version of Proposition 4.3 that suffices for the proofs in our paper. The proof of the following result is adapted from [PP89] and we provide it here only for completeness.

Proposition D.1 ([PP89]). *There exists an absolute constant $c_{\text{SORT}} > 0$ such that the following is true. For every pair of integers $m, b \geqslant 1$, there exists*

$$d_{\text{SORT}} = d_{\text{SORT}}(r, b) = c_{\text{SORT}} \cdot \log_b^2(m)$$

fixed equipartitions of $[m]$ into $\mathcal{P}_1, \ldots, \mathcal{P}_{d_{\text{SORT}}}$, each one consisting of m/b sets of size b, with the following property. Given any permutation $\sigma \in S_m$, there are d_{SORT} permutations $\gamma_1, \ldots, \gamma_{d_{\text{SORT}}}$ where for every $i \in [d_{\text{SORT}}]$, γ_i is simple on partition $\mathcal{P}_i$ so that we have $\sigma = \gamma_1 \circ \cdots \circ \gamma_{d_{\text{SORT}}}$.

This is equivalent to proving that there is a sorting network for m elements with b-comparators with depth $O(\log_b^2 m)$. Each partition $\mathcal{P}$ of $[m]$ with sets $P_1, P_2, \ldots, P_{m/b}$ of size b each can be interpreted as m/b wires as follows: wire i compares elements $x_1, x_2, \ldots x_b \in P_i$ for each $i \in [m/b]$. The permutation which is simple on $\mathcal{P}$ *only* permutes elements from P_i with each other based on how wire i arranges them.

D.1 Merge Subroutine with Large Comparators

The proof of Proposition D.1 is based on the merge sort algorithm. We assume m is a power of b for simplicity. This can be removed by padding the input with dummy elements. We start by providing a MERGE subroutine first that is used in the sorting network. For simplicity of exposition, **we assume we are working with b^2-sorters for MERGE instead of b-sorters**; we will address this easily by re-parameterizing when proving Proposition D.1.

Input: Array A which is a permutation of $[m]$ such that $A[i \cdot m/b + 1]$ to $A[(i+1) \cdot m/b]$ is in increasing order for $0 \leqslant i \leqslant (b-1)$.

Output: Array A sorted in increasing order.

Algorithm MERGE(A, m, b)**:** (with b^2-sorters instead of b-sorters)

(*i*) If $m \leqslant b^2$, compare all indices directly and sort them using a single sorter.

(*ii*) Otherwise, write array A into a $(m/b) \times b$ matrix B such that $B[i][j] = A[(i-1)b + j]$ for $1 \leqslant i \leqslant m/b$ and $1 \leqslant j \leqslant b$.

(*iii*) Run MERGE$(B[*][j], m/b, b)$ on each column $j \in [b]$ of B separately.

(*iv*) Create matrix C with dimensions $(m/b + b) \times b$ where $C[i + j - 1][j] = B[i][j]$, the empty elements in C are filled with $-\infty$ for the top right corner, and $+\infty$ for the bottom left.

For any $\ell \in [m/b^2 + 1]$, we define the **square** ℓ as all $C[i][j]$ with $(\ell - 1)b + 1 \leqslant i \leqslant \ell \cdot b$ and $1 \leqslant j \leqslant b$ (see Figure 17.)

(*v*) Divide C into m/b^2 squares of size $b \times b$ each, and sort them each using a sorter.

(*vi*) For each $\ell \in [m/b^2 - 1]$, merge the last $b^2/2$ cells of square ℓ with the first $b^2/2$ cells of square $\ell + 1$ using a single sorter for each one.

We will prove that the depth of MERGE is $O(\log_b m)$, and it works correctly. The proof of correctness is based on the standard 0-1 principle for sorting networks.

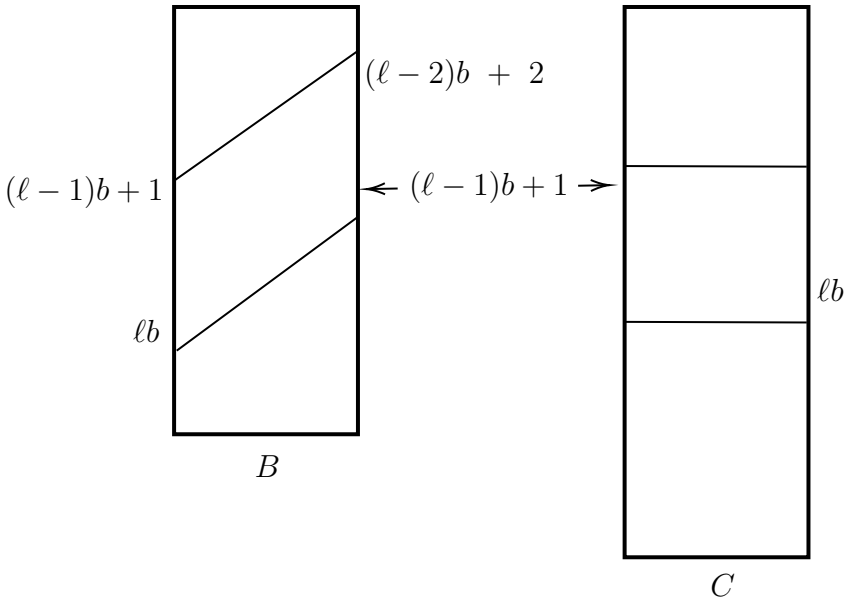


Figure 17: An illustration of a square $\ell \in [m/b^2 + 1]$ in the matrix C.

Proposition D.2 (c.f. [Knu97, Section 5.3])**.** *Any sorting network which sorts correctly when the input is from the set* $\{0,1\}^m$ *can sort any arbitrary permutation of* $[m]$ *correctly also.*

Thus, from now on we assume the input A consists of only 0's and 1's. To continue, we need a quick definition. We say a region (a region can be a row, a column or a square) **good** if it is comprised entirely of 1's or entirely of 0's. We call it a **bad** region otherwise. For any region x, we use $\mathbf{1}(x)$ and $\mathbf{0}(x)$ to denote the number of 1's and 0's in x.

Claim D.3. *At the end of step* (iii)*, each row of* B *is still sorted. Moreover, there are at most* b *bad rows in* B *and they form a contiguous region such that for each row* $i < j$ *in this region, we have* $\mathbf{1}(i) \leqslant \mathbf{1}(j)$*.*

Proof. Given that we copy each sorted region of A into m/b^2 rows of B, we have that rows of B are sorted after step (ii). Thus, for columns $p, q \in [b]$ with $p < q$, $\mathbf{1}(p) \leqslant \mathbf{1}(q)$ in matrix B at the end of step (ii). Moreover, step (iii) does not change the number of 0's and 1's in each column. After step (iii), if some row $i \in [m/b]$ of B is not sorted, there will be columns p, q such that $p < q$ but $\mathbf{1}(p) > \mathbf{1}(q)$, a contradiction. Thus, after step (iii), each row of B will still be sorted.

Let us partition the rows as follows: $P_i = \left\{j \mid (i-1) \cdot m/b^2 + 1 \leqslant j \leqslant i \cdot m/b^2\right\}$, for $i \in [b]$, namely, contiguous regions of m/b^2 rows each. After step (ii), each P_i can only have one bad row as we copied a sorted region of A into all entries in P_i. Thus, there are at most b bad rows in the matrix B overall. After step (iii), we get that these b bad rows should appear next to each other as the columns are now sorted. The sortedness also implies that for each $i < j$ in these bad rows, $\mathbf{1}(i) \leqslant \mathbf{1}(j)$ as desired. Refer to Figure 18 for an illustration. ∎

Now, with standard techniques, it is sufficient to sort every square of dimension $b \times b$ in B to get the final sorted array. However, [PP89] employs a diagonalization technique to reduce the number of squares we need to sort. The next lemma is the main reason why we do not need to sort every square of dimension $b \times b$.

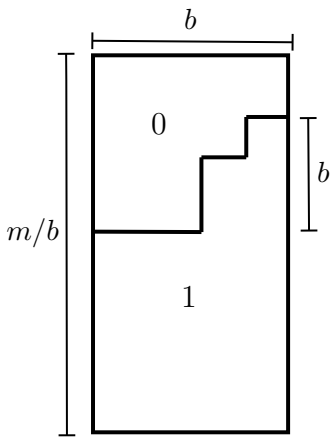


Figure 18: An illustration of matrix B after step (ii). All the cells below the partition are 1, and all the cells above it are 0.

Lemma D.4. *At the end of step (iv), for every $\ell \in [m/b^2 - 1]$: (1) if square ℓ has at least one 1, then $\mathbf{0}(\ell + 1) < b^2/2$, and (2) if square $\ell + 1$ has at least one 0, then $\mathbf{1}(\ell) < b^2/2$.*

Proof. Let square ℓ have at least one 1 in it. We will look at matrix B after step (iii) to upper bound $\mathbf{0}(\ell + 1)$. Each square ℓ in matrix C will be a rhombus in matrix B. It will contain all the cells $B[i, j]$ such that:

$$(\ell - 1)b + 1 \leqslant i + j \leqslant \ell \cdot b \quad \text{and} \quad 1 \leqslant j \leqslant b.$$

Figure 19 gives an illustration of this (and also specifies several key regions needed for the proof).

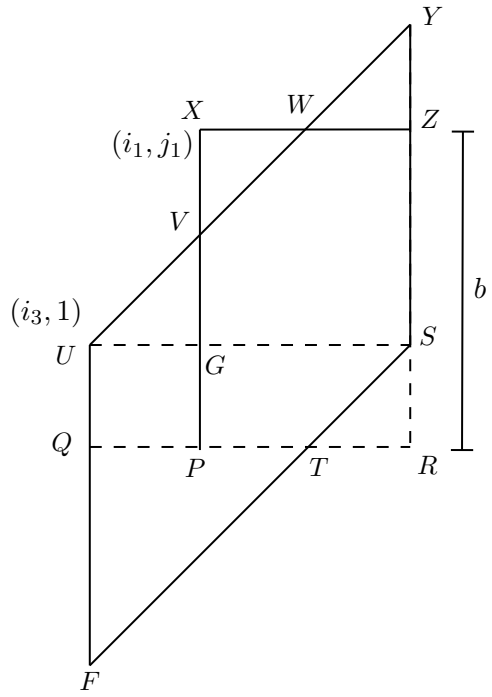


Figure 19: Upper bounding the number of 0's in square $\ell + 1$ in Lemma D.4.

Let (i_1, j_1) be the top most cell in square ℓ with a 1. In case of ties, we pick the left most

cell. By Claim D.3, we know that all the cells (i, j) with $i \geqslant i_1$ and $j \geqslant j_1$ must be a 1 in B. Let $i_2 = i_1 + b$. Cell (i_2, j_1) must be present in square $\ell + 1$ if cell (i_1, j_1) is in square ℓ.

Let $(i_3, 1)$ be the point which is the left most and bottom most cell in square ℓ (by our construction, $i_3 = \left\lceil \frac{i_1}{b} \right\rceil \cdot b$). Then, $(i_3 - j + 1, j)$ is the last cell in column j which is in square ℓ.

Refer to Figure 19 for the various possible areas in matrix B. We use $\textsc{num}(\texttt{x})$ to denote the number of cells in region x. The following regions are defined in the figure, and we make some simple observations:

- Square $\ell + 1$ is all the cells inside rhombus UYSF.
- The top most and left most 1 in square ℓ is the first cell in region XVW.
- $\textsc{num}(\texttt{XZRP}) = b(b - j_1 + 1)$.
- $\textsc{num}(\texttt{XVW}) = 1/2 \cdot (i_3 - j_1 - i_1) \cdot (i_3 - j_1 - i_1 + 1)$. (The boundary of this region is marked by the cells $(i_1, j_1), (i_1, i_3 - i_1 + 1), (i_3 - j_1 + 1, j_1)$.)
- $\textsc{num}(\texttt{STR}) = 1/2 \cdot (i_2 - i_3 - 1) \cdot (i_2 - i_3)$.
- $\textsc{num}(\texttt{QTF}) = 1/2 \cdot (i_3 + b - i_2) \cdot (i_3 + b - i_2 + 1)$.

The total number of cells with 1 in square $\ell + 1$ is lower bounded by,

$$
\begin{aligned}
\mathbf{1}(\ell + 1) &\geqslant \textsc{num}(\texttt{VWZSTP}) + \textsc{num}(\texttt{QTF}) \\
&= \textsc{num}(\texttt{XZRP}) - \textsc{num}(\texttt{XWV}) - \textsc{num}(\texttt{SRT}) + \textsc{num}(\texttt{QTF}) \\
&= b(b - j_1 + 1) - \\
&\qquad 1/2 \cdot \Big((i_3 - j_1 - i_1)(i_3 - j_1 - i_1 + 1) + (i_2 - i_3 - 1)(i_2 - i_3) - (i_3 + c - i_2)(i_3 + c - i_2 + 1) \Big) \\
&= b(b - j_1 + 1) - 1/2 \cdot ((x - j_1)(x - j_1 + 1) + (b - x)(b - x - 1) - x(x + 1)) \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{(by defining } x := i_3 - i_1) \\
&= b^2 - bj_1 + b - 1/2 \cdot \big((x - j_1)(x - j_1 + 1) + b^2 + x^2 - 2bx - b + x - x^2 - x \big) \\
&= b^2 - bj_1 + b - 1/2 \cdot \big((x - j_1)(x - j_1 + 1) + b^2 - 2bx - b \big) \\
&= 1/2 \cdot (b^2 + 3b + (x - j_1)(x - j_1 + 1 + 2b)).
\end{aligned}
$$

This quantity is minimized when $x = j_1$. Note that $i_1 + j_1 \leqslant i_3$ for cell (i_1, j_1) to belong to square ℓ, and thus, $x \geqslant j_1$. This proves one part of the lemma, as the number of 0s is upper bounded by $1/2 \cdot (b^2 - 3b)$ in square $\ell + 1$, whenever there is a cell with 1 in square ℓ.

We can prove that if there is a 0 in square $\ell + 1$, the number of 1's in square ℓ is upper bounded by $(b^2 - 3b)/2 < b^2/2$ similarly. ∎

Lemma D.5. *The network* Merge *merges the input array A correctly. Moreover, with b^2-size sorters, it has depth $O(\log_b m)$.*

Proof. By Claim D.3, there are only b contiguous bad rows in B, which implies that there are most two consecutive bad squares ℓ and $\ell + 1$ in C. By Lemma D.4, once we merge the last $b^2/2$ cells of square ℓ with first $b^2/2$ cells of square $\ell + 1$, we will definitely move all the 0's before the 1's, thus making C entirely sorted. This proves the correctness of Merge.

If we have sorters of size b^2, we can perform steps (v) and (vi) easily with 1 layer each. Let $M(m)$ denote the depth of MERGE on input size m. We get the following recurrence:

$$M(m) = M(m/b) + 2.$$

The total depth is thus $M(m) = O(\log_b m)$. ∎

D.2 The Final Sorting Network

We are now ready to give the final sorting network to sort an array of size m using MERGE primitive. For this proof, **we revert back to using b-sorters as was the original problem**.

Input: Array A which is a permutation of $[m]$.

Output: Array A sorted in increasing order.

Algorithm SORT(A)**:**

(i) If $m \leqslant b$, compare all indices directly using one b-sorter.

(ii) Otherwise, run SORT on $(A[\ell \cdot m/\sqrt{b}+1], \ldots, A[(\ell+1) \cdot m/\sqrt{b}])$ for each $0 \leqslant \ell \leqslant \sqrt{b}-1$.

(iii) Run MERGE$(A, m/\sqrt{b}, \sqrt{b})$ (with $(\sqrt{b})^2 = b$-sorters)

Proof of Proposition D.1. The correctness of primitive SORT follows from Lemma D.5, as we have b-sorters, and we merge $\sqrt{b}$ arrays of size $m/\sqrt{b}$ each. The depth of MERGE is $O(\log_{\sqrt{b}} m) = O(\log_b m)$. Let $S(m)$ be the depth of the sorting network on input of size m. Then,

$$S(m) = S(\frac{m}{\sqrt{b}}) + O(\log_b m),$$

giving us a final depth of $S(m) = O(\log_b^2 m)$ as desired. ∎