The Streaming Complexity of Cycle Counting, Sorting By Reversals, and Other Problems*

Elad Verbin † Wei Yu ‡

Abstract

In this paper we introduce a new technique for proving streaming lower bounds (and one-way communication lower bounds), by reductions from a problem called the Boolean Hidden Hypermatching problem (BHH). BHH is a generalization of the well-known Boolean Hidden Matching problem, which was used by Gavinsky et al. to prove an exponential separation between quantum communication complexity and one-way randomized communication complexity. We are the first to introduce BHH, and to prove a lower bound for it.

The hardness of the BHH problem is inherently one-way: it is easy to solve BHH using logarithmic two-way communication, but it requires $\sqrt{n}$ communication if Alice is only allowed to send messages to Bob, and not vice-versa. This one-wayness allows us to prove lower bounds, via reductions, for streaming problems and related communication problems whose hardness is also inherently one-way.

By designing reductions from BHH, we prove lower bounds for the streaming complexity of approximating the sorting by reversal distance, of approximately counting the number of cycles in a 2-regular graph, and of other problems.

For example, here is one lower bound that we prove, for a cycle-counting problem: Alice gets a perfect matching E_A on a set of n nodes, and Bob gets a perfect matching E_B on the same set of nodes. The union $E_A \cup E_B$ is a collection of cycles, and the goal is to approximate the number of cycles in this collection. We prove that if Alice is allowed to send $o(\sqrt{n})$ bits to Bob (and Bob is not allowed to send anything to Alice), then the number of cycles cannot be approximated to within a factor of 1.999, even using a randomized protocol. We prove that it is not even possible to distinguish the case where all cycles are of length 4, from the case where all cycles are of length 8. This lower bound is "natively" one-way: With 4 rounds of communication, it is easy to distinguish these two cases.

1 Introduction

Streaming algorithms are algorithms that read the input from left to right, use a small amount of space, and approximate some function of the input. Their behavior is typically measured by a tradeoff between space consumption and approximation factor. Some classical streaming problems are for example estimating frequency moments, finding approximate quantiles, and other statistics of data-sets (see [26] for more). In recent years, research has increasingly focused on streaming algorithms for estimating complex distance metrics between strings, such as earth mover distance [2], edit distance [4, 5, 3], and others. These are interesting both because of their relevance in applications, and because they are so challenging: they provide good testing grounds for exploring current techniques and coming up with new ones, both for upper bounds and for lower bounds.

Streaming lower bounds (i.e. lower bounds on the space required) often rely on reductions to communication complexity: we give Alice the first half of the input, give Bob the second half, and require them to return an answer to the problem. A lower bound on communication complexity immediately implies a lower bound on the space usage of a streaming algorithm. In fact, since the input is read left-to-right, it is enough to prove a lower bound on the one-way communication complexity, namely where Alice is only allowed to send messages to Bob, but not vice-versa (and Bob is the player who outputs the answer). However, in many lower bounds the one-wayness is never used: the communication lower bound is proved in the two-way setting, i.e. when Alice and Bob can communicate back and forth. This is a strength, not a weakness: it means that the community is proving stronger lower bounds than those actually necessary. However, considering the fact that for some problems (e.g. edit distance) the known lower bounds are exponentially far from the known upper bounds, this might arouse suspicion that we are missing techniques for proving natively one-way lower bounds. With enough understanding of one-way bounds we might gain the tools to prove stronger streaming lower bounds than those known. (Andoni and Krauthgamer explicitly discuss this point in [4], and suggest that to get better streaming lower bounds for edit distance, it might be prudent to prove "natively one-way" lower bounds for edit distance.) Indeed, the current paper focuses on problems whose lower bounds are natively one-way, in the sense that the two-way complexity of many of them is exponentially smaller than the one-way complexity.

Many classical problems, such as frequency moments, can be reduced to communication problems such as the Gap-Hamming problem (see e.g. [23, 10, 11]),

*This work was supported in part by the National Natural Science Foundation of China Grant 60553001, the National Basic Research Program of China Grant 2007CB807900, 2007CB807901.

†Aarhus University. elad.verbin@gmail.com

‡ITCS, Tsinghua University. zig.wei@gmail.com

and the SET DISJOINTNESS problem [9, 1]. For problems of estimating complicated metrics such as those discussed above, lower bound techniques seem to be much more complicated, requiring embedding arguments [3] and direct sum arguments, among others. Naturally, progress on lower bounds has been slow for those problems, and the lower bounds proved are typically logarithmic, not polynomial.

In this paper we show a technique for proving streaming lower bounds that uses one-wayness in an inherent manner, and proving nearly-linear lower bounds on the space usage. (Note that linear space is sufficient for any streaming problem since we can just store the whole input). We prove lower bounds for an approximate cycle-counting problem, for estimating the sorting by reversal distance, and for other problems. Our techniques almost, but not quite, prove lower bounds for the edit distance with block moves problem, which was studied by Cormode and Muthukrishnan [13].[1] Our lower bounds are proved by a series of reductions to a seemingly "canonical problem" (which we introduce), called BOOLEAN HIDDEN HYPERMATCHING (BHH); this problem is a variant on the well-known Boolean Hidden Matching problem [18, 19]. The reductions that we show to this problem, while non-trivial, do not require technically-complicated tools. This raises hope that there might be other results that can be proved by similar reductions, creating a new "canonical problem", along with GAP HAMMING, DISJOINTNESS, etc. The BHH problem seems to capture the hardness of the problems that we study in various natural ways, and it seems like it might capture aspects of hardness of other problems.

We proceed by describing some of the problems discussed in this paper, and the lower bounds we prove for them.

1.1 The Problems

We now describe some of the problems we consider in this paper, in their communication versions. Recall that each one-way communication lower bound immediately implies a streaming lower bound.

The Boolean Hidden Hypermatching problem has a slightly baroque definition, so we show here a simplified definition that captures the spirit of the problem. See Definition 2.5 for the actual definition.

Notice most of these are *decision problems with a promise*, in the sense that there is a promise on the inputs of Alice and Bob, which states that one of two cases hold, and the goal is always to decide which of the two cases holds. This kind of problems are particularly suitable when discussing approximation; for example they can ask: given that the value of the instance is either $\leq a$ or $\geq b$, decide between the case where it is $\leq a$ and the case where it is $\geq b$. This kind of hardness implies a hardness of approximation up to a multiplicative value of b/a, but it is conceptually more "fine-grained".

Boolean Hidden Matching (BHM) – Inaccurate Version. In this problem, Alice is given an n-bit string $x \in \{0,1\}^n$. Bob gets a perfect matching M on n vertices. Thus, the n bits of Alice are matched up in pairs, but the matching is not known to Alice. The promise is that either each matched-up pair of bits is unequal, or each matched-up pair of bits is equal. The goal is to determine which of these two cases holds.

Gavinsky et al [18, 19] proved a lower bound of $\Omega(\sqrt{n})$ for a variant of this problem called PARTIAL MATCHING. We extend that result to the BOOLEAN HIDDEN MATCHING problem in this paper.

Boolean Hidden HyperMatching (BHH) – Inaccurate Version. This is the same as the BHM problem, but where the matching M is in fact a t-uniform hypermatching. In other words, the n bits of Alice are partitioned to n/t disjoint sets of cardinality t each, and the promise is that either each matched-up set of bits XORs to 1, or each matched-up set of bits XORs to 0. The goal is to determine which of these two cases holds.

For this problem, we prove a lower bound of $\Omega(n^{1-1/t})$. The proof is similar to that of [19] but we need to generalize various aspects of the proof to deal with hypermatchings and the stronger promise.

Cycle Counting – Gap Version. Alice gets a perfect matching E_A on a bipartite graph with n vertices on each side, and Bob gets a prefect matching E_B on the same graph. The union $E_A \cup E_B$ is a collection of disjoint cycles. They wish to approximate the number of cycles in the union. The goal is to decide between the case that the number of cycles is $\leq a$ and the case where the number of cycles is $\geq b$.

We consider the special case of this problem where $b = 2a$ and where b divides n (e.g. $b = n/2, n/3, \ldots$). We prove a lower bound of $\Omega\left((n/2)^{1-b/n}\right)$ on the randomized one-way communication complexity. This implies, for example, that when we wish to decide between the case that there are $\leq n/4$ cycles and the case there are $\geq n/2$ cycles, the communication complexity is $\Omega(\sqrt{n})$. This lower bound is proved by a reduction to the BOOLEAN HIDDEN HYPERMATCHING problem.

[1] We mention this since the edit distance with block moves problem, suggested to us by Robert Krauthgamer, was the original motivation for our work. We have not proved a lower bound on it, but we do believe that the techniques in the current paper can eventually prove a lower bound for this problem as well.

Deterministic variants of this problem was studied by Raz [27] and by Harvey [22], but the deterministic case is very different from the randomized case. In particular, it is easy to see that a simple hashing-based protocol can estimate to excellent accuracy the number of cycles of length 2, yielding a protocol that distinguishes the case of $\leq 0.1n$ cycles from the case of $\geq 0.9n$ cycles with $O(1)$ communication and probability of success 0.999. Such a protocol probably does not exist in the deterministic setting.

Sorting by Reversals. The input is a signed permutation x of $\{1, \ldots, n\}$: this is a permutation where each element is also assigned a sign of plus or minus. Alice gets the first half of this permutation (i.e. the first $n/2$ elements), and Bob gets the second half. They wish to estimate the *reversal distance*, namely the smallest number of *reversals* required to transform x to the positive identity permutation, $(+1, +2, \ldots, +n)$. A *reversal* is the operation of choosing a block of the permutation x, reversing the order of the elements and flipping the signs.

There is a known (and rather complicated) polynomial-time algorithm that given x computes exactly the reversal distance of x [20]. There is even a linear-time algorithm that achieves this [6]. However, here we are interested in the communication complexity (or streaming complexity) of approximating the reversal distance.

For this problem, we prove a lower bound of $\Omega((n/8)^{1-1/t})$ on the randomized one-way communication complexity of getting a $(1 + 1/(4t - 2) - \epsilon)$-multiplicative approximation of the reversal distance (for any $\epsilon > 0$). For example, to get a 1.166-approximation, $\Omega(\sqrt{n})$ communication is required. To get a 1.0001-approximation, $\Omega(n^{0.999})$ communication is required.

We also prove lower bounds for sorting by block interchanges and a few other problems. Furthermore, we discuss the problems of sorting by transpositions and of edit distance with block moves, but we do not prove lower bounds for them.

One observation to note is that BHH is qualitatively different than sorting by reversals or cycle counting, in that it is highly non-symmetric: the role of Alice is entirely different than the role of Bob. Another way to express this idea is that for the BHH problem, if Bob is allowed to send one message to Alice *instead* of Alice sending a message to Bob, the BHH problem becomes easy ($t \log n$ bits of communication suffice). This does not hold for the other problems: in the other problems, if Bob is allowed to send one message to Alice, the lower bounds stay the same.

The structure of our reductions is depicted in Figure 1.

In Section 5 we outline some ideas behind this work and pose some conjectures that might motivate future work. The reader might wish to skim it *before* moving on.

2 Main Results

In this section we prove that Sorting-by-Reversal and Sorting-by-Block-Interchanges in the streaming model requires space $\Omega((n/8)^{1-1/t})$ to achieve approximation factor $1 + 1/(4t - 2)$. That is, in order to achieve $1 + \epsilon$ approximate, a lower bound of $\Omega((n/8)^{\frac{1-2\epsilon}{1+2\epsilon}})$ is required.

2.1 List of Applications The main problem we investigate is the Sorting-by-Reversal problem in the streaming model. We also talk about Sorting-by-Block-Interchanges and Sorting-by-Transpositions problem. However, we do not have a formalized lower bound for the Sorting-by-Transpositions problem.

Definition 2.1. (Sorting-by-Reversal) *Given a data stream of a permutation S on $\{1, \ldots, n\}$ where each coordinate of S is also assigned with a sign of plus or minus. We define a reversal $r(i, j)$ on x transforming $x = (x_1, \ldots, x_n)$ to*

$$x' = (x_1, \ldots, x_{i-1}, -x_j, -x_{j-1}, \ldots, -x_i, x_{j+1}, \ldots, x_n).$$

In the Sorting-by-Reversal *problem we want to return the minimum number of reversals $sbr(S)$ in order to sort S (i.e. transform S into $(1, 2, \ldots, n)$) within $1+\epsilon$ multiplicative factor by scanning the data stream in one pass.*

Definition 2.2. (Sorting-by-Block-Interchanges) *Given a string of permutation x, we define a block-interchange $r(i, j, k, l)$ where $1 \leq i \leq j < k \leq l \leq n$ on x transforming $x = (x_1, \ldots, x_n)$ to*

$$x' = (x_1, \ldots, x_{i-1}, x_k, x_{k+1}, \ldots, x_l, x_{j+1}, \ldots, x_{k-1}, x_i, x_{i+1}, \ldots, x_j, x_{l+1}, \ldots, x_n).$$

The goal of Sorting-by-Block-Interchanges *is to approximate the minimum number of block-interchanges to sort x in increasing order (i.e. $1, 2, \ldots, n$) by scanning one pass on the input.*

The theorems giving lower bounds are Theorem 2.3 and Theorem 2.2 respectively.

$$\text{Boolean Hidden Hypermatching} \to \text{Gap Cycle-Counting} \to \begin{cases} \text{Sorting-by-Reversal} \\ \text{Sorting-by-Block-Interchange} \end{cases}$$

Figure 1: Reduction

Definition 2.3. (Sorting-by-Transpositions) *Given a string of permutation x, we define a transposition $t(i,j,k)$ where $1 \le i \le j < k \le n$ on x transforming $x = (x_1, \ldots, x_n)$ to*

$$x' = (x_1, \ldots, x_{i-1}, x_{j+1}, \ldots, x_k, \\ x_i, x_{i+1}, \ldots, x_j, x_{k+1}, \ldots, x_n).$$

The goal of Sorting-by-Transpositions *is to approximate the minimum number of transpositions to sort x in increasing order by scanning one pass on the input.*

A polynomial approximation algorithm in the non-streaming model was proposed in [12]. We do not have a lower bound for the SBT problem in this paper but we believe it is closely related to our result (see Conjecture 5.3).

2.2 Cycle Counting and Boolean Hidden Hypermatching Problem In this subsection we define the Gap Cyc-Counting (abbr. GCC) problem used in the reduction to get the lower bound for above listed problems. The GCC problem is a communication complexity problem with parameters a and b as following.

Definition 2.4. ($\mathrm{GCC}_n(a,b)$) *Let $G = (U \cup V, E_A \cup E_B)$ to be a bipartite graph with U and V being the vertices on two sides respectively where $|U| = |V| = n$. Alice receives E_A, a perfect matching in G; and Bob receives E_B, another perfect matching in G. In the* Gap Cycle-Counting *problem it is promised that the number of cycles in G is either $\le a$ or $\ge b$. They want to*

- *return 0, when the number of cycles in $G \le a$;*
- *return 1, when the number of cycles in $G \ge b$.*

Note that this problem could also be defined as deciding the number of cycles is either $\le a$ or $\ge b$ in the product uv of two permutations $u, v \in S_n$ where S_n is the symmetric group of order n.

We reduce the GCC problem from the Boolean Hidden Hypermatching problem, which is a variant of the Partial Matching problem proposed in [19], which is actually a generalization of the Boolean Hidden Matching [8] problem.

Definition 2.5. (BHH_n^t) *The* Boolean Hidden Hypermatching *problem is a communication complexity problem where Alice gets a boolean vector $x \in \{0,1\}^n$ where $n = 2kt$ for some integer k, and Bob gets a perfect hypermatching M on n vertices where each edge has t vertices and a boolean vector w of length n/t. Let Mx denote the length-n/t boolean vector $(\bigoplus_{1\le i\le t} x_{M_{1,i}}, \ldots, \bigoplus_{1\le i\le t} x_{M_{n/t,i}})$ where $(M_{1,1}, \ldots, M_{1,t}), \ldots, (M_{n/t,1}, \ldots, M_{n/t,t})$ are the edges of M. It is promised that either $Mx \oplus w = \vec{\mathbf{1}}$ or $Mx \oplus w = \vec{\mathbf{0}}$. The problem is to return 1 when $Mx \oplus w = 1^{n/t}$, and 0 when $Mx \oplus w = 0^{n/t}$.*

The main theorem of this paper is the following lower bound for BHH_n^t.

Theorem 2.1. *The randomized one-way communication complexity of BHH_n^t when $n = 2kt$ for some integer $k \ge 1$ is $\Omega(n^{1-1/t})$.*

This theorem is proved in Section 3. This lower bound is the best possible up to a log factor because of a birthday paradox upper bound: Alice can just send $O(n^{1-1/t})$ random indices from x and with good probability Bob will get all of the values of one of the hyperedges. For deterministic and one-sided error protocols we can get linear lower bounds; we omit their proofs here.

Lemma 2.1. *If there is a randomized one-way protocol for $GCC_{2n}(n/t, 2n/t)$, then there is a randomized one-way protocol for BHH_n^t problem on n vertices using the same communication and error probability.*

Proof. Consider an instance of the BHH_n^t problem: Alice gets a boolean vector x of length n, and Bob gets a hypermatching M of n/t edges where each edge is of size t, as well as a boolean vector w of length n/t.

We now construct an instance of the GCC problem. It is a bipartite graph $(U \cup V, E_A \cup E_B)$, where $U = \{u_1, \ldots, u_{2n}\}$ and $V = \{v_1, \ldots, v_{2n}\}$. Here, E_A and E_B are Alice's and Bob's input respectively which are perfect matchings in the bipartite graph. For each $i \in [n]$, we place the following edges in E_A:

- If $x_i = 0$ then place $(u_{2i-1}, v_{2i-1}) \in E_A$ and $(u_{2i}, v_{2i}) \in E_A$; we call this a parallel gadget;

- If $x_i = 1$ then place $(u_{2i-1}, v_{2i}) \in E_A$ and $(u_{2i}, v_{2i-1}) \in E_A$; we call this a cross gadget.

That is, a "parallel gadget" for $x_i = 0$, or a "cross gadget" for $x_i = 1$.

Then for each hyperedge $(i_1, i_2, \ldots, i_t) \in M$,

- For $k = 1, 2, \ldots, t-1$, $(u_{2i_k-1}, v_{2i_{k+1}-1}) \in E_B$ and $(u_{2i_k}, v_{2i_{k+1}}) \in E_B$;
- For $k = t$, if $w_i = 0$, we just let $(u_{2i_t-1}, v_{2i_1-1}), (u_{2i_t}, v_{2i_1}) \in E_B$; if $w_i = 1$, we let $(u_{2i_t-1}, v_{2i_1}), (u_{2i_t}, v_{2i_1-1}) \in E_B$. (The latter case means that we add an extra "cross gadget" if $w_i = 1$.)

It is easy to see that if $|\{k|x_{i_k} = 1\}| = p$, then we go through $p+w_i$ cross gadgets when traversing from u_{2i_1-1} along the cycle. Thus, if $p+w_i$ is odd then there will be one cycle of length $4t$, otherwise there will be two cycles of length $2t$. So if the correct result for BHH_n^t is 1, we know that for any hyperedge i, $p + w_i$ is always odd. It means that the number of cross gadgets is odd, so each hyperedge will form two cycles of length $2t$, which means the number of cycles is $2n/t$. If the result is 0, by a similar argument the number of cycles will be n/t.

A randomized protocol for $\mathrm{GCC}(n/t, 2n/t)$ on $4n$ vertices will, with probability $1 - \epsilon$, return 0 if the number of cycles is $\leq n/t$ and will return 1 if the number of cycles is $\geq 2n/t$. So for an input from BHH_n^t we can turn it into two matchings in the above way. After that we can invoke the protocol for $\mathrm{GCC}_{2n}(n/t, 2n/t)$ vertices to get a protocol for BHH_n^t.

By using Lemma 2.1 with Theorem 2.1, we obtain the following lower bound for Gap Cycle-Counting problem.

Corollary 2.1. *The randomized one-way communication complexity with error probability* 1/100 *of* $GCC_n(n/2t, n/t)$ *for any integer* $t|n$ *is* $\Omega((n/2)^{1-1/t})$.

Note that this reduction actually suffers from a loss in the hardness of the GCC problem. The GCC problem is hard no matter if the one-way communication is from Alice to Bob or from Bob to Alice. However, the BHH problem is easy if Bob speaks to Alive, since Bob could just send one edge using $t \log n$.

2.3 Hardness of Sorting-by-Reversal and Sorting-by-Block-Interchange We already see the hardness of the Gap Cycle-Counting (abbr. GCC) problem, and we are going to show that the Sorting-by-Reversal (abbr. SBR) and Sorting-by-Block-Interchanges (abbr. SBI) problems in the streaming model are also hard. We do this by relating the number of cycles in the *breakpoint graph* of the permutation to the number of cycles in the input of GCC.

Definition 2.6. (Breakpoint Graph) *For a signed permutation* S *of* $[n]$[2]*, the breakpoint graph* G *of* S *is constructed by the following algorithm.*

- *From* S *we construct another integer array* T *of length* $2n + 2$.
  - *If* $S[i] > 0$, $T[2i] = 2|S[i]| - 1$, $T[2i + 1] = 2|S[i]|$.
  - *If* $S[i] < 0$, $T[2i] = 2|S[i]|$, $T[2i + 1] = 2|S[i]| - 1$.
  - $T[1] = 0$, $T[2n + 2] = 2n + 1$.
- $G = ([2n + 2], E)$, *where* E *is constructed as following.*
  - *For* $1 \leq i \leq n + 1$, *let* $(2i - 1, 2i) \in E$ *and we call these edges black edges.*
  - *If* $|T[j] - T[k]| = 1$ *and* $\forall 1 \leq i \leq n, \{j, k\} \neq \{2i, 2i + 1\}$, *let* $(j, k) \in E$ *and we call these gray edges.*

In short, we expand an integer in S to two according to its sign, after that we insert 0 at the front and $2n+1$ to the rear. After that, we connect every other pairs and every pair that the value only differs by one but not expanded from the same integer. The breakpoint is a standard and useful tool to get SBR and SBI distance from the number of cycles. For further understanding on the breakpoint graph (sometimes also called as *cycle graph*), the reader might want to read [20, 17].

Lemma 2.2. *Let* E_A *and* E_B *to be two perfect matchings on bipartite graph of size* $2n$ *(*n *vertices on each side). And let* C *to be the number of cycles in* $E_A \cup E_B$. *We can transform* E_A *to an integer array* S_1 *and* E_B *to another integer array* S_2, *such that* $S = S_1 + S_2$ *is a signed permutation of* $[4n]$ *which has* $2C + 1$ *cycles in its breakpoint graph. Moreover, let* S_1' *and* S_2' *satisfy* $S_1'[i] = |S_1[i]|$ *and* $S_2'[i] = |S_2[i]|$, *then* S' *satisfying* $S'[i] = |S[i]|$ *is an unsigned permutation of* $[4n]$ *which also has* $2C + 1$ *cycles in its breakpoint graph.*

Before proving the lemma, we first show how to get the lower bound of SBI by using this lemma.

[2]We use $[n]$ to denote $\{1, 2, \ldots, n\}$.

THEOREM 2.2. *The lower bound of the space in the streaming model for* SORTING-BY-BLOCK-INTERCHANGE *when we want to approximate the correct distance up to multiplicative factor* $1 + 1/(4t-2) - \epsilon$ *for any small constant* ϵ *is* $\Omega((n/8)^{1-1/t})$.

Proof. Assuming the space used in the streaming algorithm for computing the SORTING-BY-BLOCK-INTERCHANGE distance here is $\mathcal{S}$, we can cut the stream in the middle and split it into two sets of numbers of size $n/2$. Thus by the standard reduction we can turn the streaming algorithm into a one-way communication complexity protocol, where Alice holds the first part of the stream and the Bob holds the second part of the stream. After Alice simulates her input with the streaming algorithm, she can send $\mathcal{S}$ bits to Bob which is the memory sketch of the algorithm, and Bob can resume the algorithm after that to get the correct answer. Thus, if we can give a lower bound for this one-way communication complexity version of the problem, there will be a lower bound for $\mathcal{S}$.

Say in the $\mathrm{GCC}_{n/4}(n/4t, n/8t)$ problem Alice receives a matching E_A in a bipartite graph of $n/2$ vertices ($n/4$ vertices on each side) and Bob receives another matching E_B and they want to decide the number of cycles in $E_A \cup E_B$ is either $\geq n/4t$ or $\leq n/8t$. We can transform E_A and E_B to S_1' and S_2' by using Lemma 2.2. Let C to be the number of cycles in $E_A \cup E_B$, then we know that the number of cycles in the breakpoint graph of $S_1' + S_2'$ is $2C+1$. We also know that the SBI distance of $S' = S_1' + S_2'$ satisfies $sbi(S') = (n-2C)/2$ because of the following lemma from [12].

LEMMA 2.3. (THEOREM 4 FROM [12]) *The* SORTING-BY-BLOCK-INTERCHANGE *distance for a permutation* x *of* $\{1, 2, \ldots, n\}$ *is*

$$sbi(x) = \frac{n+1-c(x)}{2}$$

where $c(x)$ *is the number of cycles in the breakpoint graph of* x.

Assuming we have a streaming algorithm for approximating the SBI distance using space $\mathcal{S}$, by the above discussion we can turn it into a one-way communication complexity protocol for approximating the SBI distance of S' by using $\mathcal{S}$ bits where Alice holds S_1' and Bob holds S_2'.

However, if this protocol can approximate up to $1 + 1/(4t-2) - \epsilon$ for some integer $t \geq 2$, then we can distinguish SBI distance $n - n/2t$ from distance $n - n/4t$ since $(1 + 1/(4t-2) - \epsilon) \cdot (n - n/2t) < n - n/4t$. Thus it means that a protocol that approximates SBI distance within $1 + 1/(4t-2) - \epsilon$ will be able to distinguish the $C \leq n/4t$ case and the $C \geq n/2t$ case by simply comparing $(n - 2sbi(S'))/2$ with $n/2t$. By using Corollary 2.1 we have a lower bound of $\Omega((n/8)^{1-1/t})$ for approximating SBI distance on an array of length n with factor $1 + 1/(4t-2) - \epsilon$ for any constant $\epsilon > 0$.

We are expecting to repeat the same proof for SBR. But this does not work straight forward because we do not have things like Lemma 2.3 for SBR. We can only have the following conditional lemma for SBR from [20].

LEMMA 2.4. (THEOREM 4 FROM [20]) *If no cycles in the breakpoint graph* G *are unoriented, then the optimal number of reversals is* $n - c$, *where* c *is the number of cycles in the breakpoint graph.*

And the definition for oriented cycles is as following.

DEFINITION 2.7. (ORIENTED EDGES AND CYCLES) *Say a cycle* i_1, i_2, ..., i_k *where* i_1 *is the smallest vertex in the cycle is unoriented if* $k > 2$ *and* $i_1 < i_2 < \ldots < i_k$. *Otherwise, we call the cycle oriented. It could be observed that in the breakpoint graph a cycle is oriented iff it has length* 2, *or there is a gray edge* (i, j) *in the cycle that both* i *and* j *are left ends (or both right ends) of black edges.*

Fortunately, we can prove that the breakpoint graph constructed in our reduction only has oriented cycles.

LEMMA 2.5. *All cycles in the breakpoint graph of the signed permutation constructed in Lemma 2.2 are oriented.*

So by combining this lemma with Lemma 2.4 we can have the lower bound for SBR by repeating the same proof in Theorem 2.2 with SBR.

THEOREM 2.3. *The lower bound of the space in the streaming model for* SORTING-BY-REVERSAL *when we want to approximate the correct answer up to multiplicative factor* $1 + 1/(4t-2) - \epsilon$ *for any small constant* ϵ *is* $\Omega((n/8)^{1-1/t})$, *where* n *is the number of integers in the stream.*

Proof. Using S instead of S', and using Lemma 2.5 and Lemma 2.4 instead of Lemma 2.3, it is easy to see we can repeat the same proof for 2.2 to get the same lower bound.

2.4 Proof of Lemma 2.2 and Lemma 2.5 There are two lemmas still left unproved, we prove them here.

LEMMA 2.6. (LEMMA 2.2 RESTATED) *Let* E_A *and* E_B *to be two perfect matchings on bipartite graph of size* $2n$

(n vertices on each side). And let C to be the number of cycles in $E_A \cup E_B$. We can transform E_A to an integer array S_1 and E_B to another integer array S_2, such that $S = S_1 + S_2$ is a signed permutation of $[4n]$ which has $2C+1$ cycles in its breakpoint graph. Moreover, let S'_1 and S'_2 satisfy $S'_1[i] = |S_1[i]|$ and $S'_2[i] = |S_2[i]|$, then S' that will satisfy $S'[i] = |S[i]|$ is an unsigned permutation of $[4n]$ which also has $2C+1$ cycles in its breakpoint graph.

Proof. Starting with E_A and E_B as perfect matchings, we construct S as following.

- For each edge $(u_i, v_j) \in E_A$, $S_1[2j] = 2j$ and $S_1[2j-1] = -(2i+2n)$. Since E_A is a permutation, all the indices in $\{1, \ldots, 2n\}$ are assigned.
- For each edge $(u_i, v_j) \in E_B$, $S_2[2i-1] = 2n+2i-1$ and $S_2[2i] = -(2j-1)$.

The breakpoint graph $G = (P, E)$ of S could be constructed according to Definition 2.6, which is also illustrated in Figure 2.

We are going to show the number of cycles in the breakpoint graph G is $2C+1$, where C is the number of cycles $E_A \cup E_B$.

Let $u_{i_1} \to v_{j_1} \to \cdots \to u_{i_m} \to v_{j_m} \to u_{i_1}$ be a cycle of length m in G. And we know that $(u_{i_k}, v_{j_k}) \in E_A$ and $(u_{i_{k+1}}, v_{j_k}) \in E_B$. We are going to show there are two corresponding cycles in G, say $4n+4i_1-1 \to 4j_1-4 \dashrightarrow 4j_1-3 \to 4n+4i_2+1 \dashrightarrow \cdots \to 4n+4i_1-2 \dashrightarrow 4n+4i_1-1$ and $4n+4i_1 \to 4j_1-1 \dashrightarrow 4j_1-2 \to 4n+4i_2-2 \dashrightarrow \cdots \to 4n+4i_1+1 \dashrightarrow 4n+4i_1$ where $\to$ denotes black edge and $\dashrightarrow$ denotes gray edge. Note that the direction here are just for the convenience of presentation, the cycles are actually undirected. This could be observed by the following facts.

- $(4j_1-4, 4n+4i_1-1)$ and $(4n+4i_1, 4j_1-1)$ are black edges. This is because $(u_{i_1}, v_{j_1}) \in E_A$, so either $2j_1-2, -(2n+2i_1), 2j_1$ are consecutive integers in S or $j_1 = 1$ and $-(2n+2i_1), 2$ are consecutive integers in S.
- $(4j_1-1, 4j_1-2)$ and $(4j_1-4, 4j_1-3)$ are gray edges.
- $(4j_1-3, 4n+4i_2+1)$ and $(4j_1-2, 4n+4i_1-4)$ are black edges. This is because $(u_{i_2}, v_{j_1}) \in E_B$, so either $2n+2i_2-1, -(2j_1-1), 2n+2i_2+1$ are consecutive integers in S or $i_2 = n$ and $4n, -(2j_1-1)$ are the last integers in S.
- $(4n+4i_2-2, 4n+4i_1-3)$ and $(4n+4i_2+1, 4n+4i_2)$ are gray edges.

All these cycles in $E_A \cup E_B$ will use $8n$ edges in all. There are only two edges left. But there is another cycle which is $4n+1 \to 4n+2 \dashrightarrow 4n+1$ of length 2 which are just formed by the edges left. So the number of cycles in G is $2C+1$, where C is the number of cycles in $E_A \cup E_B$. (And, the reader could also observe that all the cycles in G has twice the length as the cycles in $E_A \cup E_B$.)

For the cycles in the breakpoint graph of S' the proof is similar. We omit the details and refer the reader to (d) in Figure 2.

LEMMA 2.7. (LEMMA 2.5 RESTATED) *All cycles in the breakpoint graph of the signed permutation constructed in Lemma 2.2 are oriented.*

Proof. According to the construction all the gray edges except $(4n+2, 4n+1)$ are of the kind $(4j_1-1, 4j_1-2)$,$(4j_1-4, 4j_1-3)$, $(4n+4i_2-2, 4n+4i_1-3)$ and $(4n+4i_2+1, 4n+4i_2)$. And since $4j_1-1, 4j_1-2, 4n+2i_1+1, 4n+4i_1$ are all right ends of black edges, and $4n+4i_2-2, 4n+4i_1-3, 4j_1-4, 4j_1-3$ are all left ends of black edges, we have that all these cycles are oriented. We refer the reader to (c) in Figure 2.

And the cycle $4n+1 \to 4n+2 \dashrightarrow 4n+1$ is just a length 2 cycle, which is also oriented. Thus it means the SBR distance is $4n+1-(2C+1) = 4n-2C$.

3 Lower bound of BHH

In this section we prove a lower bound on the BHH^t_n problem. Together with the reductions in the previous sections, it gives the main result of the paper.

3.1 Notations A t-dimensional hypermatching M is a set of vertex disjoint edges, and each edge is a set of t vertices. In this paper for a given boolean vector x we are in particular interested in the XOR value of the edges on this vector. That is, if the edges of M are $(i_{1,1}, \ldots, i_{1,t}), \cdots, (i_{n/t,1}, \ldots, i_{n/t,t})$, we are interested in values $\bigoplus_{1 \le k \le t} x_{i_{1,k}}, \cdots, \bigoplus_{1 \le k \le t} x_{i_{n/t,k}}$. We use Mx to denote this vector.

Another way of representing the hypermatching M is a $n/t \times n$ matrix. Each row of the matrix has t 1's and $n-t$ 0's denote the corresponding position of an edge. The advantage of this representation is that when computing the XOR values of all the edges on vector x in the matching, it is equivalent to performing the matrix-vector multiplication Mx. Thus we abuse notation and use M both for the hypermatching and the matrix. Note that in the matrix multiplication here, all computations are modulo 2.

In the $t = 2$ case M is a perfect matching on n vertices. This was the case discussed in [19].

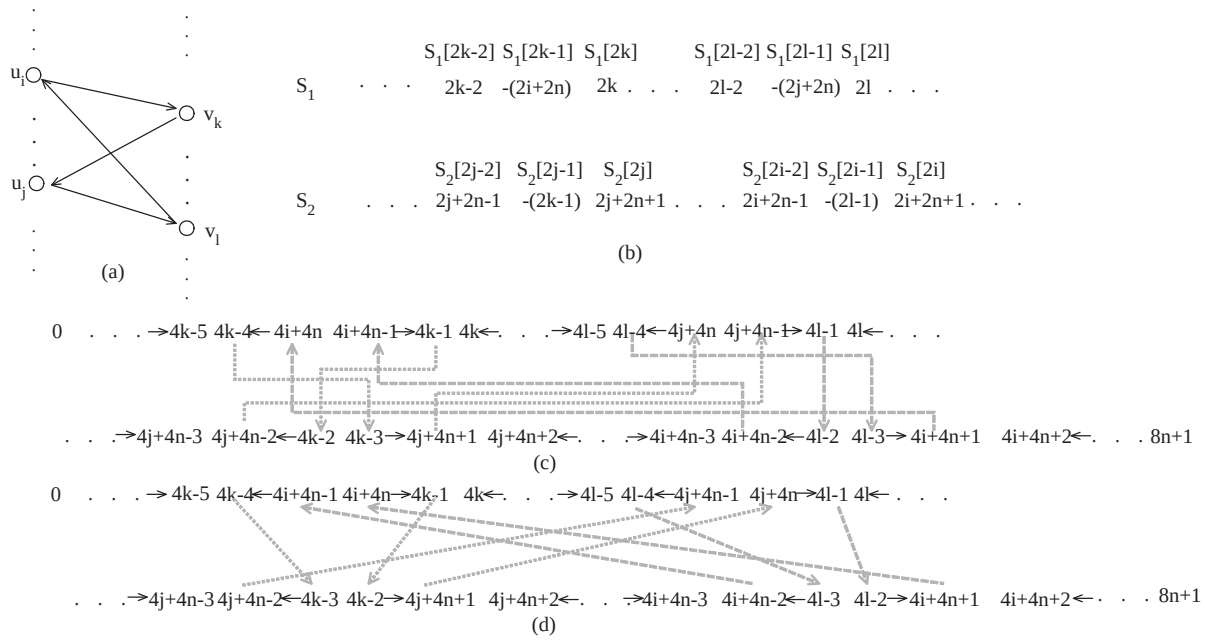

Figure 2: Example of building the breakpoint graph. Dashed edges are gray edges while solid ones (horizontal ones) are black edges. (a) An cycle in $E_A \cup E_B$. (b) Corresponding integers in S_1 and S_2. (c) The two corresponding cycles in the breakpoint graph. (d) The breakpoint graph in the unsigned case. Dashed lines are gray edges and the solid ones are black edges. Note: all orientations are for the convenience of presentation, all edges are undirected in fact.

For two distributions D and D', let $\Delta(D, D')$ denote the total variance distance between D and D', which is $\sum_x |D(x) - D'(x)|$.

3.2 Lower bound of BHH We now prove a lower bound for the BHH_n^t problem (see Definition 2.5). The proof here extends the proof in [19] to t-hypermatching case when $n = 2kt$. We are using the same framework and notations as in [19].

The main difference here is we need to talk about *perfect hypermatching* instead of partial matching. In order to show a lower bound for the *perfect* matching, we need to think of the case that $n = 2kt$ and talk about *single parity* sets (for definition see below). For extending to hypermatching, we need to recompute all the parameters used in [19]. The reader may want to read [19] to get a better understanding of the proof.

The basic idea of the proof is the following. First, we apply Yao's minimax principle, after that we can talk about deterministic protocols under distributional inputs. Alice's message is short, so after Bob looks at her message, there are typically still many possible inputs of Alice that he cannot tell apart. In other words, let l to be the length of the message sent from Alice to Bob, then the number of inputs for which Alice sends a specific message is typically 2^{n-l}, which is a large number. Let X be a r.v. uniformly distributed in a set of size $\geq 2^{n-l}$ that corresponds to one specific message from Alice to Bob. We are going to show that for most of the hypermatchings M, if 2^{n-l} is large, the distribution of MX is nearly the same as $\overline{MX}$, which means Bob cannot distinguish these two cases. So l must be large to make such kind of sets small. Note that unlike [19] here MX and $\overline{MX}$ are close to each other, but neither of them is close to uniform distribution.

Definition 3.1. (Single Parity Set) *We call a set $A \subseteq \{0,1\}^n$ a single parity set iff $\exists c \in \{0,1\}$, $\forall x \in A$, $|x| \equiv c \pmod 2$, i.e., all the elements in A have the same weight parity.*

THEOREM 3.1. *Let $n = 2pt$ for some integer p, and x be uniformly distributed over a* single parity *set $A \subseteq \{0,1\}^n$ of size $|A| \geq 2^{n-l}$ for some $l \geq 1$, and let M be uniformly distributed over the set $\mathcal{M}_{n,t}$ of all t-hypermatchings on n vertices. There exists a universal constant $\gamma > 0$ (independent of n, l, t, ϵ), such that for all $\epsilon \in (0,1]$: if $l \leq \gamma \epsilon n^{1-1/t}$ then*

$$\mathop{\mathbb{E}}_{M \in M_{n,t}} [\Delta(p_M, q_M)] \leq \epsilon,$$

where p_M and q_M are the distributions over $\{0,1\}^{n/t}$ whose p.d.f. are

$$p_M(z) = \frac{|\{x | x \in A, Mx = z\}|}{|A|},$$

and

$$q_M(z) = \frac{|\{x | x \in A, Mx = \overline{z}\}|}{|A|},$$

respectively.

It is important n needs to be an even multiple of t, otherwise there is a simple constant upper bound by sending the parity of x to Bob. We leave the proof of this theorem in Section 4, we will show in detail how to get the lower bound of BHH_n^t from it.

THEOREM 3.2. (THEOREM 2.1 RESTATED) *A one-way communication complexity protocol that correctly computes BHH^t with error probability $\leq 1/100$ has at least $\Omega(n^{1-1/t})$ bits.*

Proof. Think of a protocol with error $\leq \epsilon$. By Yao's principle, there must be a deterministic protocol which has distributional error $\leq \epsilon$ under the following chosen "hard distribution". We choose Alice's input X and Bob's input M independently and uniformly at random; then we choose $w = Mx$ with probability $1/2$ and $w = \overline{Mx}$ with probability $1/2$.

Let Π be the transcript of the protocol. In our case, Π is just one message, of length $\leq l$, sent from Alice to Bob. The transcript can be thought of splitting Alice's input into disjoint single parity sets $A_1, A_2, \ldots, A_{2^{l+1}}$, where each set contains all the inputs for which she sends the same message and *single parity*. Consider all such sets bigger than $2^n/(100 \cdot 2^{l+1})$. It is trivial to see that least 99% of the universe is covered by such sets. Therefore, with probability 0.99, the message sent from Alice to Bob is from a set of size $\geq 2^n/(100 \cdot 2^{l+1})$.

Let such a set to be A, and let X be a random variable uniformly distributed on A, and let $Z = MX$ for some M which is a prefect t-hypermatching on n vertices. Let $Z_0 = Z$ and $Z_1 = Z \oplus 1^{n/t}$. If $l \leq \gamma/100 \cdot n^{1-1/t} - \log 100 - 1$, then $|A| \geq 2^{\gamma/100 \cdot n^{1-1/t}}$ and we know A is single parity. Thus we can use Theorem 3.1 to get $\mathbb{E}_M [\Delta(Z_0, Z_1)] < 1/100$. By Markov bound, for at least $1 - 1/10$ fraction of all M, $\Delta(Z_0, Z_1) \leq 1/10$.

For all the M in that case, by looking at w Bob needs to decide which distribution w is from. That is, Bob needs to distinguish two distributions Z_0 and Z_1. Since it is well known that the best protocol to distinguish 1/10-close (i.e. $\Delta(Z_0, Z_1) \leq 1/10$) distributions errs with probability $\geq 1/2 - 1/40$ (see also [19, Eq 1.1]). So we know that the total error will be $\geq 9/10 \cdot (1/2 - 1/40) \cdot 99\% > 1/100$, which contradicts the assumption. Thus $l > \Omega(n^{1-1/t})$ and we are done.

4 Proof for Theorem 3.1

In this section we prove Theorem 3.1.

4.1 Preliminaries on Fourier Analysis Here we need the standard definitions [15] of discrete Fourier analysis. Let $f : \{0,1\}^n \to \mathbb{R}$ be a function on the boolean cube. We define the inner product on boolean functions by

$$\langle f, g \rangle = \mathop{\mathbb{E}}_{x \in \{0,1\}^n} [f(x)g(x)] = \mathbb{E}[f \cdot g] \ .$$

This also defines the ℓ_2-norm

$$\|f\|_2 = \sqrt{\langle f, f \rangle} = \sqrt{\mathbb{E}[f^2]}.$$

The Fourier transform of f is a function $\hat{f} : \{0,1\}^n \to \mathbb{R}$ defined by

$$\hat{f}(s) = \langle f, \chi_s \rangle = \mathop{\mathbb{E}}_{y \in \{0,1\}^n} [f(y)\chi_s(y)]$$

where $\chi_S : \{0,1\}^n \to \mathbb{R}$ is the character function $\chi_S(y) = \prod_{i \in S}(2y_i - 1)$. For the sake of convenience, we also use s to denote the characteristic boolean vector of s, e.g. χ_s can also be defined by $\chi_s(y) = (-1)^{y \cdot s}$. We will need Pareseval's identity.

LEMMA 4.1. (PARSEVAL) *For every function $f : \{0,1\}^n \to \mathbb{R}$ we have $\|f\|_2^2 = \sum_{x \subseteq [n]} \hat{f}(s)^2$.*

And we need the following lemma, which is a direct consequence of the Bonami-Gross-Beckner inequality, or hypercontractivity inequality from KKL [24]. Let $|x|$ denote the Hamming weight of x, i.e., number of 1's in the boolean vector x.

LEMMA 4.2. (KKL [24]) *Let f be a function $f : \{0,1\}^n \to \{-1, 0, 1\}$. Let $A = \{x | f(x) \neq 0\}$. Then for every $\delta \in [0,1]$ we have*

$$\sum_{s \in \{0,1\}^n} \delta^{|s|} \hat{f}(s)^2 \leq \left(\frac{|A|}{2^n}\right)^{\frac{2}{1+\delta}} .$$

4.2 Main Proof Think of any set $A \subseteq \{0,1\}^n$ with $|A| \geq 2^{n-l}$. Let f be its characteristic function (i.e. $f(x) = 1$ iff $x \in A$), and x be uniformly distributed on A. The theorem defines the following functions for $z \in \{0,1\}^{n/t}$:

$$p_M(z) = \frac{|\{x|x \in A, Mx = z\}|}{|A|},$$

and

$$q_M(z) = \frac{|\{x|x \in A, Mx = \overline{z}\}|}{|A|}.$$

CLAIM 4.1. *For function $f : \{0,1\}^n \to \{0,1\}$ which is an indicator function of a single parity set $A \subseteq \{0,1\}^n$ (i.e. $f(x) = 1 \iff x \in A$), we have*

$$\hat{f}(v)^2 = \hat{f}(\overline{v})^2.$$

Proof. First, assume that $\forall x \in A$, $|x| \equiv 0 \pmod 2$, then we know that $\forall v \in \{0,1\}^n$, $\forall x \in A$, $v \cdot x = \overline{v} \cdot x$. Thus the following equation holds,

$$\begin{aligned}\hat{f}(v) &= \frac{1}{2^n} \sum_{x \in \{0,1\}^n} f(x)(-1)^{v \cdot x} \\ &= \frac{1}{2^n} \sum_{x \in \{0,1\}^n} f(x)(-1)^{\overline{v} \cdot x} \\ &= \hat{f}(\overline{v}).\end{aligned}$$

For the $\forall x \in A$, $|x| \equiv 1 \pmod 2$ case it could be shown in a similar fashion that $\hat{f}(v) = -\hat{f}(\overline{v})$.

We prove Theorem 3.1 here.

Proof. By Jensen's inequality we know that

$$\mathop{\mathbb{E}}_{M}[\Delta(p_M, q_M)] \leq \sqrt{\mathop{\mathbb{E}}_{M}[\Delta(p_M, q_M)^2]},$$

then by Cauchy-Schwarz inequality we have

$$\mathop{\mathbb{E}}_{M}\left[\Delta(p_M, q_M)^2\right] \leq 2^{2n/t} \mathop{\mathbb{E}}_{M}\left[\|p_M - q_M\|_2^2\right].$$

So by Parseval 4.1 and let $r_M(z) = p_M(z) - q_M(z)$ we have

$$2^{2n/t} \mathop{\mathbb{E}}_{M}\left[\|p_M - q_M\|_2^2\right] = 2^{2n/t} \mathop{\mathbb{E}}_{M}\left[\sum_{s \in \{0,1\}^{n/t}} \widehat{r_M}(s)^2\right].$$

We are going to use the Fourier coefficients of f to represent the Fourier coefficients of r_M. After that we classify the Fourier coefficients into two parts: 1) One part is decaying fast, we bound these by using KKL (Lemma 4.2); 2) the other part decays slow, but they are already small, we bound them directly.

Let $z \cdot s$ denote the inner product of z and s (note that all the plus operations are done in the mod 2 base). Then we can write $\widehat{r_M}$ as following

$$\begin{aligned}\widehat{r_M}(s) &= \frac{1}{2^{n/t}} \left(\sum_{s \subseteq \{0,1\}^{n/t}} p_M(z)(-1)^{z \cdot s} \right. \\ &\qquad \left. - \sum_{z \in \{0,1\}^{n/t}} q_M(z)(-1)^{z \cdot s} \right) \\ &= \frac{1}{|A| \cdot 2^{n/t}} \left(\sum_{s \subseteq \{0,1\}^{n/t}} p_M(z)(-1)^{z \cdot s} \right. \\ &\qquad \left. - \sum_{z \in \{0,1\}^{n/t}} p_M(z)(-1)^{\overline{z} \cdot s} \right).\end{aligned}$$

Since we know that $(-1)^{z \cdot s + \overline{z} \cdot s} = (-1)^{|s|}$, so when $|s|$ is even we have $z \cdot s = \overline{z} \cdot s$ and when $|s|$ is odd we have $z \cdot s = 1 - \overline{z} \cdot s$. Thus we know when $|s|$ is even $\widehat{r_M}(s) = 0$. For s satisfying $|s|$ is odd,

$$\begin{aligned}\widehat{r_M}(s) &= \frac{2}{|A| \cdot 2^{n/t}} \sum_{z \in \{0,1\}^{n/t}} p_M(z)(-1)^{z \cdot s} \\ &= \frac{2}{|A| \cdot 2^{n/t}} \left(|\{x \in A | (Mx) \cdot s = 0\}| \right. \\ &\qquad \left. - |\{x \in A | (Mx) \cdot s = 1\}| \right) \\ &= \frac{2}{|A| \cdot 2^{n/t}} \left(|\{x \in A | x \cdot (M^T s) = 0\}| \right. \\ &\qquad \left. - |\{x \in A | x \cdot (M^T s) = 1\}| \right) \\ &= \frac{2}{|A| \cdot 2^{n/t}} \sum_{x \in \{0,1\}^n} f(x)(-1)^{x \cdot (M^T s)} \\ &= \frac{2^{n+1}}{|A| \cdot 2^{n/t}} \cdot \hat{f}(M^T s).\end{aligned}$$

Thus,

$$
\begin{aligned}
&2^{2n/t}\mathop{\mathbb{E}}_{M}\left[\sum_{s\in\{0,1\}^{n/t}}\widehat{r_M}(s)^2\right]\\
&=\frac{2^{2n+2}}{|A|^2}\mathop{\mathbb{E}}_{M}\left[\sum_{\substack{s\in\{0,1\}^{n/t}\\ |s| \text{ odd}}}\hat{f}(M^Ts)^2\right]\\
&=\frac{2^{2n+2}}{|A|^2}\mathop{\mathbb{E}}_{M}\left[\sum_{\substack{v\in\{0,1\}^{n}\\ |v|=kt\\ k \text{ odd}}}\Big|\{s\in\{0,1\}^{n/t}|M^Ts=v\}\Big|\cdot\hat{f}(v)^2\right]\\
&=\frac{2^{2n+2}}{|A|^2}\sum_{\substack{v\in\{0,1\}^{n}\\ |v|=kt\\ k \text{ odd}}}\Pr_{M}\Big[\exists s\in\{0,1\}^{n/t}\text{s.t.}M^Ts=v\Big]\cdot\hat{f}(v)^2.
\end{aligned}
$$

It could be easily observed that for a fixed k, $\Pr_M\left[\exists S\in\{0,1\}^{n/t}\text{s.t.}M^TS=v\right]$ are the same for different v. So let $g(k)=\Pr_M\left[\exists S\in\{0,1\}^{n/t}\text{s.t.}M^TS=v\right]$, we have the following claim to be proved later.

Claim 4.2. *Let $x\in\{0,1\}^n$ such that $|x|=kt$. We define*

$$
g(k)\triangleq\Pr_M\Big[\exists z\in\{0,1\}^{n/t}\ s.t.\ M^Tz=x\Big]=\frac{\binom{n/t}{k}}{\binom{n}{kt}},
$$

where M is uniformly over all t-hypermatchings on n vertices. Thus $g(k)=g(n/t-k)$ and

$$
g(k)\le\frac{(en/t/k)^k}{(n/t/k)^{kt}}=e^k\cdot(kt)^{kt-k}\cdot n^{k-kt}.
$$

Since we know that

$$
2^{2n/t}\mathop{\mathbb{E}}_{M}\left[\sum_{s\in\{0,1\}^{n/t}}\widehat{r_M}(s)^2\right]=\frac{2^{2n+2}}{|A|^2}\sum_{\substack{1\le k\le n/t\\ k \text{ odd}}}g(k)\sum_{\substack{v\in\{0,1\}^n\\ |v|=kt}}\hat{f}(v)^2.
$$

Because A is a single parity set, we have $\hat{f}(v)^2=\hat{f}(\overline{v})^2$ by Claim 4.1. And since $g(k)=g(n/t-k)$ and $2t|n$ from the statement of the theorem, we have,

$$
\begin{aligned}
&\frac{2^{2n+2}}{|A|^2}\sum_{\substack{1\le k\le n/t\\ k \text{ odd}}}g(k)\sum_{\substack{v\in\{0,1\}^n\\ |v|=kt}}\hat{f}(v)^2\\
&\quad=\frac{2^{2n+3}}{|A|^2}\sum_{\substack{1\le k\le n/2t\\ k \text{ odd}}}g(k)\sum_{\substack{v\in\{0,1\}^n\\ |v|=kt}}\hat{f}(v)^2.
\end{aligned}
$$

We are going to bound this value by two parts.

Part I ($1\le kt<4l$). For a fixed k, let $\delta=kt/4l$, we have that

$$
\begin{aligned}
\text{(Claim 4.2)}\qquad g(k)&\le e^k\cdot(kt)^{kt-k}\cdot n^{k-kt}\\
&=e^k\cdot\delta^{kt}\cdot\frac{(4\gamma\epsilon)^{kt}}{(kt)^k}\\
&\le\frac{\epsilon^{kt}}{32}\cdot\delta^{kt}
\end{aligned}
$$

where the last inequality holds because $k\ge1$, $t\ge2$ and by choosing γ to be a sufficiently small constant (e.g. 1/1024, independent of k, t).

Thus for a fixed k we have

$$
\begin{aligned}
\frac{2^{2n+3}}{|A|^2}g(k)\sum_{v:|v|=kt}\hat{f}(v)^2&\le\frac{2^{2n+3}\cdot\epsilon^{kt}}{32\cdot|A|^2\cdot2^{kt}}\sum_{v:|v|=kt}\delta^{kt}\hat{f}(v)^2\\
\text{(Lemma 4.2)}\qquad&\le\frac{\epsilon^{kt}}{4}\cdot\frac{2^{2n}}{|A|^2}\cdot\left(\frac{|A|}{2^n}\right)^{\frac{2}{1+\delta}}\\
&\le\frac{\epsilon^{kt}}{4}\cdot\frac{2^{2n}}{|A|^2}\cdot\left(\frac{|A|}{2^n}\right)^{2}\\
&=\epsilon^{kt}/4.
\end{aligned}
$$

Summing up, for $\epsilon<1/2$, we have

$$
\begin{aligned}
\frac{2^{2n+3}}{|A|^2}\sum_{\substack{k=1\\ k \text{ odd}}}^{4l/t}g(k)\sum_{v:|v|=kt}\hat{f}(v)^2&\le\sum_{k=1}^{4l}\epsilon^{kt}/4\\
&\le\epsilon^2/2.
\end{aligned}
$$

Part II ($4l\le kt\le n/2$). We know that

$$
\begin{aligned}
\frac{g(k-1)}{g(k)}&=\frac{\binom{n/t}{k-1}/\binom{n}{(k-1)t}}{\binom{n/t}{k}/\binom{n}{kt}}\\
&=\prod_{i=1}^{t-1}\frac{n-kt+i}{kt-t+i}\\
&\ge1
\end{aligned}
$$

when $k\le n/2t$. Since by Parseval (Lemma 4.1) we have $\sum_z\hat{f}(z)^2=\frac{|A|}{2^n}$ and we know that $\frac{2^n}{|A|}\le2^l$,

$$
\frac{2^{2n}}{|A|^2}\sum_{\substack{4l\le kt\le n/2\\ k \text{ odd}}}g(k)\sum_{z:|z|=kt}\hat{f}(z)^2\le2^l\cdot g(4l/t).
$$

As we know

$$
\begin{aligned}
2^lg(4l/t)&\le2^l\cdot e^{4l/t}\cdot(4l)^{4l-4l/t}\cdot n^{4l/t-4l}\\
&\le\left(\frac{2e^{4/t}(4\gamma)^{4-4/t}}{\left(n^{1/t}\right)^{1-1/t}}\cdot\epsilon^{4-4/t}\right)^l\\
&\le\epsilon^2/2
\end{aligned}
$$

where the last step is because $t \geq 2$, $l \geq 1$ and γ to be a sufficiently small constant (e.g. $1/1024$).

By summing up the two parts we conclude that

$$\begin{aligned}
&2^{2n/t} \mathop{\mathbb{E}}_{M} \left[\|p_M - q_M\|_2^2\right] \\
&= 2^{2n/t} \mathop{\mathbb{E}}_{M} \left[\sum_{s\in\{0,1\}^{n/t}} \widehat{r_M}(s)^2\right] \\
&= \frac{2^{2n+3}}{|A|^2} \sum_{\substack{1\leq k\leq n/2t \\ k \text{ odd}}} g(k) \sum_{\substack{v\in\{0,1\}^n \\ |v|=kt}} \hat{f}(v)^2 \\
&\leq \frac{\epsilon^2}{2} + \frac{\epsilon^2}{2} = \epsilon^2.
\end{aligned}$$

Thus,

$$\begin{aligned}
\mathop{\mathbb{E}}_{M} \left[\Delta(p_M, q_M)\right] &\leq \sqrt{\mathop{\mathbb{E}}_{M} \left[\Delta(p_M, q_M)^2\right]} \\
&= \sqrt{2^{2n/t} \mathop{\mathbb{E}}_{M} \left[\|p_M - q_M\|_2^2\right]} \leq \epsilon.
\end{aligned}$$

At last we prove Claim 4.2 left in the proof.

Proof. We can think of $x = 1^{kt}0^{n-kt}$ w.l.o.g. M here contains n/t edges with size t each. Each edge needs to be either a subset of $[kt]$ or a subset of $[n] \setminus [kt]$. And notice that there is at most one s for a fixed z to make $M^T s = z$ true. So the number of choosing k of them in $[kt]$ and $n/t - k$ of them in $[n] \setminus [kt]$ is

$$\frac{(kt)!}{(t!)^k (k)!} \cdot \frac{(n-kt)!}{(t!)^{n/t-k}(n-n)!(n/t-k)!}.$$

By dividing the number of all possible matchings $n!/((t!)^{n/t}(n/t)!(n-n)!)$ we have

$$\frac{\frac{(kt)!}{(t!)^k (k)!} \cdot \frac{(n-kt)!}{(t!)^{n/t-k}(n-n)!(n/t-k)!}}{(n!/((t!)^{n/t}(n/t)!(n-n)!)} = \frac{\binom{n/t}{k}}{\binom{n}{kt}}$$

which is called $g(k)$ as a function of k.

5 Ideas Behind this Work and Suggestions for Further Work

The approach taken in this paper does not match the authors' original plan. In this section we outline the original thoughts and plans. For future research it might be interesting to use some ideas from these original plans, or even to try to follow them more closely than we did. This might make the results simpler, more elegant, or quantitatively stronger.

The original motivation of this work has been to prove a $n^{1-\epsilon}$ lower bound on the communication complexity of edit distance with block moves (viz. Conjecture 5.3). Edit distance with block moves is a metric where the distance between two strings x, y is the minimum number of operations required to transform x to y by single-character edit operations (i.e. insertion, deletion and modification of a character) and by block moves (i.e. cutting out an entire block and inserting it elsewhere). Alice gets x, Bob gets y, and they want to approximate the distance between x and y. For this problem, there is a protocol with communication $O(1)$ and approximation factor $O(\log n \log^* n)$, see [13], which works by an embedding into ℓ_1. In the case that each character appears exactly once in x and exactly once in y, i.e. each string is a permutation, this problem can be called "Ulam distance with block moves", and it is easy to obtain an protocol with communication $O(1)$ and which achieves $2+\epsilon$ approximation, by a simple embedding into ℓ_1, as in [14]. We believe that approximating these problems better than some particular constant factor $c > 1$ should take at least $\sqrt{n}$ communication. (1.5 and 2 are both good guesses for the value of c.) We also believe that the closer the approximation factor is to 1, the closer the communication must be to n. Thus, we conjecture for example that for a 1.001-approximation, the communication must be $n^{0.99}$. Such lower bounds would in particular imply strong lower bounds on the space complexity of approximating edit distance with block moves in the streaming model, a well-studied problem.

The problem of Ulam distance with block moves is equivalent to a well-known problem in computational biology called Sorting-by-Transpositions (SBT) [7, 21, 16]. This problem is neither known to be NP-hard nor known to lie in P. Therefore, we decided to start by considering a related problem, Sorting-by-Reversal (SBR), which is known to lie in P. For this problem, Hannenhalli and Pevzner [20, 25] proved a combinatorial characterization of the distance. This combinatorial characterization heavily relies on the so-called *breakpoint graph* (see Definition 2.6) which is a 2-regular graph. The distance is "almost" uniquely determined by the number of cycles in this graph. (We say "almost" because the number of "hurdles" and "fortresses" also needs to be taken into account, see [20, 25] for more details.) Furthermore, when we are trying to compute the edit distance between two signed permutations x and y using the Hannenhalli-Pevzner characterization, the breakpoint graph is a union of two matchings, one of which is determined by x and the other is determined by y. We see then that the communication complexity of sorting by reversals is tightly related to the communication complexity of cycle-counting, at least at the conceptual level. This led us to study the communication complexity of the cycle counting problem.

For the cycle counting problem, we had originally

made the following conjecture:

CONJECTURE 5.1. *For the cycle-counting problem, it takes $\Omega(\sqrt{n})$ communication to distinguish the case where all cycles are of length 4 from the case that there is just one cycle of length $2n$. In general, it takes $\Omega(n^{1-1/\ell})$ communication to distinguish the case where all cycles are of length 2ℓ from the case that there is just one cycle of length $2n$.*

We tried to prove this conjecture directly, but failed. Instead, we found that cycle counting reduces to BHH by a rather straightforward reduction, and decided to work on a lower bound for BHH. This reduction together with the lower bound for BHH that we prove, gives Corollary 2.1, which is a weaker result than Conjecture 5.1. For example, Corollary 2.1 tells us that it takes $\Omega(\sqrt{n})$ communication to distinguish the case that all cycles are of length 4 from the case that all cycles are of length 8, while Conjecture 5.1 tells us that it takes $\Omega(\sqrt{n})$ communication to distinguish the case that all cycles are of length 4 from the case that there is just one cycle, of length $2n$.

Back to SBR: First let us ignore the issue of hurdles and fortresses. In this case, if two permutations x, y induce a breakpoint graph which has only cycles of length 4, then the SBR distance between x and y is roughly $n/2$; on the other hand, when x and y induce a breakpoint graph that consists of just one very long cycle, then the SBR distance between x and y is roughly n. Therefore, assuming Conjecture 5.1, it is tempting to conjecture that any protocol which approximates SBR to within $2-\epsilon$ needs $\sqrt{n}$ communication. Since we can only prove Corollary 2.1, then we should instead consider the case that x and y induce a breakpoint graph which has only cycles of length 8; in this case, the SBR distance between x and y is roughly $3n/4$. Thus, surely we can prove that to approximate SBR to within $3/2-\epsilon$, one needs $\sqrt{n}$ communication? But this is not the case. The issue of dealing with hurdles and fortresses, and the fact that not any 2-regular graph (with particular vertex labels) is the breakpoint graph of some pair of permutations, prevented us from proving this. We were able to find a more baroque reduction that shows that to approximate SBR to within $7/6-\epsilon$, $\sqrt{n}$ communication is needed, this reduction is described in Theorem 2.3. Overall, we think our difficulties were just of technical nature and we still believe our original intuitions to be true. Therefore we conjecture:

CONJECTURE 5.2. *Any communication protocol for approximating SBR or SBI up to $2-\epsilon$ requires $\Omega(\sqrt{n})$ communication. Furthermore, for any number α of the form $\alpha = 2/1, 3/2, 4/3, 5/4, \ldots$, any protocol to approximate SBR or SBI up to $\alpha - \epsilon$ requires $\Omega(n^{1/\alpha})$ communication.*

For SBT we have no good guess what the exact tradeoff should be between the approximation factor and the communication required, but in general the behavior should be similar: the closer the approximation factor is to 1, the closer the communication should be to n.

CONJECTURE 5.3. *For any $\gamma > 0$ there exists a $\delta > 0$ such that any protocol for approximating SBT up to $1+\delta$ requires $\Omega(n^{1-\gamma})$ communication.*

The difficulty behind dealing with SBT, and the reason we were not able to prove any lower bounds about it, is that no combinatorial characterization of the SBT distance is known. However, we believe that our baroque reduction from SBR uses the combinatorial characterization to an excessive degree, and that a better reduction might be adaptable to SBT. For this the main difficulty is in the fact that not every 2-regular graph (with particular vertex labels) can be obtained as the breakpoint graph of some pair of permutations.

Finally, all of the discussion so far has been about one-way protocols. BHH is already easy with 2 rounds of communication, but for cycle counting, SBR, SBT, etc, the problems seem hard even for 2 rounds or more. In particular, for cycle counting we make the following conjecture.

CONJECTURE 5.4. *For protocols where $\leq t$ messages are sent, communication $\Omega(n^{1-1/t})$ is needed to distinguish the case where all cycles are of length $2t$ and the case where all cycles are of length $4t$.*

Acknowledgement

We are indebted to Robert Krauthgamer for suggesting to work on lower bounds for edit distance with block moves, on invaluable advice, including referring us to the Boolean Hidden Matching problem, and for a lot of encouragement along the way. We would also like to thank Alexandr Andoni for many helpful discussions on streaming lower bounds.

References

[1] N. Alon, Y. Matias, and M. Szegedy. The space complexity of approximating the frequency moments. *Journal of Computer and system sciences*, 58(1):137–147, 1999.

[2] A. Andoni, K. Do Ba, P. Indyk, and D. Woodruff. Efficient sketches for earth-mover distance, with applications. In *2009 50th Annual IEEE Symposium on Foundations of Computer Science*, pages 324–330. IEEE, 2009.

[3] A. Andoni, T.S. Jayram, and M. Patrascu. Lower bounds for edit distance and product metrics via Poincaré-type inequalities. In *ACM-SIAM Symposium on Discrete Algorithms (SODA10)*, 2010.

[4] A. Andoni and R. Krauthgamer. The Computational Hardness of Estimating Edit Distance [Extended Abstract]. In *Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science*, pages 724–734. IEEE Computer Society, 2007.

[5] A. Andoni, R. Krauthgamer, and K. Onak. Polylogarithmic Approximation for Edit Distance and the Asymmetric Query Complexity. *Arxiv preprint arXiv:1005.4033*, 2010.

[6] D.A. Bader, B.M.E. Moret, and M. Yan. A linear-time algorithm for computing inversion distance between signed permutations with an experimental study. *Journal of Computational Biology*, 8(5):483–491, 2001.

[7] V. Bafna and P.A. Pevzner. Sorting by Transpositions. *SIAM Journal on Discrete Mathematics*, 11(2):240, 1998.

[8] Z. Bar-Yossef, T.S. Jayram, and I. Kerenidis. Exponential Separation of Quantum and Classical One-Way Communication Complexity. *SIAM Journal on Computing*, 38:366, 2008.

[9] Z. Bar-Yossef, T.S. Jayram, R. Kumar, and D. Sivakumar. An information statistics approach to data stream and communication complexity. *Journal of Computer and System Sciences*, 68(4):702–732, 2004.

[10] J. Brody, A. Chakrabarti, O. Regev, T. Vidick, and R. de Wolf. Better Gap-Hamming Lower Bounds via Better Round Elimination. *Arxiv preprint arXiv:0912.5276*, 2009.

[11] A. Chakrabarti and O. Regev. An optimal lower bound on the communication complexity of gap-hamming-distance. *ECCC Report TR10-140*, 2010.

[12] D.A. Christie. Sorting permutations by block-interchanges. *Information Processing Letters*, 60(4):165–169, 1996.

[13] G. Cormode and S. Muthukrishnan. The string edit distance matching problem with moves. *ACM Transactions on Algorithms (TALG)*, 3(1):1–19, 2007.

[14] G. Cormode, S. Muthukrishnan, and C. Sahinalp. Permutation editing and matching via embeddings. In *Automata, languages and programming: 28th international colloquium, ICALP 2001, Crete, Greece, July 8-12, 2001: proceedings*, page 481. Springer Verlag, 2001.

[15] R. De Wolf. A brief introduction to Fourier analysis on the Boolean cube. *Theory of Computing Library–Graduate Surveys*, 1:1–20, 2008.

[16] I. Elias and T. Hartman. A 1.375-approximation algorithm for sorting by transpositions. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, pages 369–379, 2006.

[17] J. Feng and D. Zhu. Faster algorithms for sorting by transpositions and sorting by block interchanges. *ACM Transactions on Algorithms (TALG)*, 3(3):25, 2007.

[18] D. Gavinsky, J. Kempe, and R. de Wolf. Exponential separation of quantum and classical one-way communication complexity for a boolean function. *Electronic Colloquium on Computational Complexity*, TR06-086, July 2006.

[19] D. Gavinsky, J. Kempe, I. Kerenidis, R. Raz, and R. de Wolf. Exponential Separation for One-Way Quantum Communication Complexity, with Applications to Cryptography. *SIAM Journal on Computing*, 38:1695, 2008.

[20] S. Hannenhalli and P.A. Pevzner. Transforming cabbage into turnip: polynomial algorithm for sorting signed permutations by reversals. *Journal of the ACM (JACM)*, 46(1):1–27, 1999.

[21] T. Hartman. A simpler 1.5-approximation algorithm for sorting by transpositions. In *Combinatorial pattern matching*, pages 156–169. Springer, 2003.

[22] N.J.A. Harvey. Matroid intersection, pointer chasing, and Young's seminormal representation of S n. In *Proceedings of the nineteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 542–549. Society for Industrial and Applied Mathematics, 2008.

[23] T.S. Jayram, R. Kumar, and D. Sivakumar. The One-Way Communication Complexity of Hamming Distance. *Theory Of Computing*, 4:129–135, 2008.

[24] J. Kahn, G. Kalai, and N. Linial. The influence of variables on Boolean functions. In *Proceedings of the 29th Annual Symposium on Foundations of Computer Science*, pages 68–80. IEEE Computer Society, 1988.

[25] H. Kaplan, R. Shamir, and R.E. Tarjan. A Faster and Simpler Algorithm for Sorting Signed Permutations by Reversals. *SIAM Journal on Computing*, 29:880, 2000.

[26] S. Muthukrishnan. *Data streams: Algorithms and applications*. Now Publishers Inc, 2005.

[27] R. Raz and B. Spieker. On the log rank-conjecture in communication complexity. *Combinatorica*, 15(4):567–588, 1995.

A Variants of BHM

In this section we discuss the variants of Boolean Hidden Matching. In short, our main conclusion is they are all equivalent up to a constant factor in the amount of bits communicated.

Definition A.1. (BHM_n^1) *The Variant 1 of the* Boolean Hidden Matching *problem is a communication complexity problem where*

- *Alice holds a boolean vector* $x \in \{0,1\}^n$,
- *Bob holds a perfect matching* M *on* n *vertices and a boolean vector* w *of length* $n/2$.

It is promised that either $Mx \oplus w = 1^{n/2}$ *or* $Mx \oplus w = 0^{n/2}$. *The problem is to return* 1 *for* $Mx \oplus w = 1^{n/2}$ *case, and* 0 *for* $Mx \oplus w = 0^{n/2}$ *case.*

DEFINITION A.2. (BHM_n^2) *The Variant 2 of the* BOOLEAN HIDDEN MATCHING *problem is a communication complexity problem where*

- *Alice holds two boolean vectors* $x \in \{0,1\}^n$ *and* $y \in \{0,1\}^{n/2}$,
- *Bob holds a perfect matching* M *on* n *vertices and a boolean vector* w *of length* $n/2$.

It is promised that either $Mx \oplus y \oplus w = 1^{n/2}$ *or* $Mx \oplus y \oplus w = 0^{n/2}$. *The problem is to return* 1 *for* $Mx \oplus y \oplus w = 1^{n/2}$ *case, and* 0 *for* $Mx \oplus y \oplus w = 0^{n/2}$ *case.*

DEFINITION A.3. (BHM_n^3) *The Variant 2 of the* BOOLEAN HIDDEN MATCHING *problem is a communication complexity problem where,*

- *Alice hold a boolean vector* $x \in \{0,1\}^n$,
- *Bob holds a bipartite matching (could also be seen as a permutation)* M *on* $2n$ *vertices (*n *on each side) and a boolean vector* w *of length* n,

It is promised that either $M(x) \oplus x \oplus w = \mathbf{1}^n$ *or* $M(x) \oplus x \oplus w = 0^n$. *The problem is to return* 1 *for* $M(x) \oplus x \oplus w = 1^n$ *case, and* 0 *for* $M(x) \oplus x \oplus w = 0^n$ *case.*

DEFINITION A.4. (BHM_n^{1s}, BHM_n^{2s} AND BHM_n^{3s}) *BHM_n^{1s}, BHM_n^{2s}, and BHM_n^{3s} are exactly the same problem as BHM_n^1, BHM_n^2, and BHM_n^3 problem respectively with* w *fixed to be* $0^{n/2}$.

LEMMA A.1. (EQUIVALENCE OF THE 6 VARIANTS) *All the 6 variants of the problem stated above are all equivalent in the sense that the communication complexity are within constant factors of each other.*

Proof. It is easy to see that BHM_n^{1s}, BHM_n^{2s} and BHM_n^{3s} are just special cases of BHM_n^1, BHM_n^2, and BHM_n^3. Moreover, BHM_n^1 is a special case of BHM_n^2 by setting $y = 0^n$.

$\text{BHM}_{2n}^1 \Rightarrow \text{BHM}_n^3$. Let $x' = xx$ to be the concatenation of two identical copies of x and M' to be the perfect matching on x' which is built from M that connects the first half of x' to the second half of x', that is, connecting i to $M(i) + n/2$. By executing the protocol for BHM_{2n}^1 on x', M' and w we will get the answer.

$\text{BHM}_{2n}^3 \Rightarrow \text{BHM}_n^2$. Take the inputs x, y, M, w of BHM_n^2. Let $x' = xy$ and $w' = ww$. For each edge $(i,j) \in M$, we add two edges $(i, j+n)$ and $(i+n, j)$ to the bipartite matching M'. After that we run the protocol for BHM_{2n}^3 on x', M', w' to get the answer.

$\text{BHM}_{2n}^{1s} \Rightarrow \text{BHM}_n^1$. Take the inputs x, M, w of BHM_n^1. Let $x' = x\overline{x}$, where $\overline{x}$ means the bitwise negation vector of x. And for each edge $(i_k, j_k) \in M$, if $w_k = 0$ we let $(i_k, j_k), (i_k + n, j_k + n) \in M'$, otherwise when $w_k = 1$ we let $(i_k, j_k + n), (i_k + n, j_k) \in M'$. After that we run the protocol for BHM_{2n}^{1s} on x', M' to get the answer.

We know that $\text{BHM}_{4n}^{1s} \Rightarrow \text{BHM}_{2n}^{3s} \Rightarrow \text{BHM}_n^{2s}$ by using similar reduction. This finishes the proof.