Secure Multiparty Computation in the Bounded Storage Model

Jiahui Liu* and Satyanarayana Vusirikala*

*The University of Texas at Austin
jiahui@cs.utexas.edu, satya.vus@gmail.com

Abstract. Most cryptography is based on assumptions such as factoring and discrete log, which assume an adversary has bounded computational power. With the recent development in quantum computing as well as concern with everlasting security, there is an interest in coming up with information-theoretic constructions in the bounded storage model.

In this model, an adversary is computationally unbounded but has limited space. Past works have constructed schemes such as key exchange and bit commitment in this model. In this work, we expand the functionalities further by building a semi-honest MPC protocol in the bounded storage model. We use the hardness of the parity learning problem (recently shown by Ran Raz (FOCS 16) without any cryptographic assumptions) to prove the security of our construction, following the work by Guan and Zhandry (EUROCRYPT 19).

1 Introduction

Many schemes in cryptography rely on various computational assumptions such as factoring, discrete log and Learning with Errors. Even though these assumptions are well-believed to be true, recent advances in quantum computing show that factoring and discrete log problems could be solved efficiently by quantum computers [24,11]. It, therefore, could be dangerous to base the security of global user information on these assumptions. An adversary can store ciphertext and attempt to decrypt it later when his computational power increases or quantum computers come into existence. The current systems, therefore, do not provide everlasting security. Alternately, one could construct schemes in the information-theoretic model, where no computational power of the adversary is assumed. However, many of these information-theoretic systems such as one-time pad are impractical to use.

In the face of the above issues, Maurer proposed bounded storage model [18] in which we do not assume any computational restrictions on the adversary. Rather, we assume that the adversary has bounded storage and is unable to store a long conversation. In this model, [18] constructed a key exchange protocol assuming a publicly accessible long random stream of bits. In his protocol, Alice and Bob respectively record a private random subset of n bits from a stream of n^2 random bits. They later send their recorded positions to each other, and the

secret key is set as the bit at their commonly recorded position in the stream. Note that they will record at least one-bit position in common with a constant probability according to the birthday paradox. An eavesdropping adversary with Cn^2 (for some constant $C < 1$) storage can only obtain the secret key with $1/C$ probability.

Many sequence of works [18,3,2,1,4,7,17,8,22,13,19,23,6] have given increasingly secure and efficient protocols for key exchange, oblivious transfer, commitments and timestamping in this model. Most of these works rely on the birthday paradox. Unfortunately, this has several disadvantages. For example, in the above protocol, (1) The honest parties can succeed with only constant probability. To achieve high success probability, the protocol has to be repeated several times. (2) The adversary can succeed with constant probability. To achieve statistical security, a randomness extractor has to be applied to the obtained secret key. (3) The birthday paradox does not have a rich structure that can be exploited to construct advanced protocols.

Recently, Ran Raz et al. [20,21,16] proposed a new class of techniques that can be used to construct cryptographic schemes in the bounded storage model. Specifically, they studied the hardness of solving *parity learning* problem in bounded space. In this problem, a secret string k is sampled uniformly at random from $\{0,1\}^n$. A learner has to compute the secret k when given a stream of samples $(a_1, b_1), (a_2, b_2), \cdots$, where each a_i is sampled uniformly at random from $\{0,1\}^n$, and b_i is the inner product of a_i and k mod 2. Roughly speaking, [20] proved that any (computationally unbounded) learner that uses less than $n^2/20$ space requires either an exponential number of samples or has an exponentially small probability of outputting the correct answer. [20] used the hardness of solving the parity learning problem to construct a simple secret key encryption protocol – The secret key k is randomly sampled from $\{0,1\}^n$. To encrypt a bit b, sample $x \leftarrow \{0,1\}^n$ and output $(x, x \cdot k + b \bmod 2)$. In this protocol, the honest users require only $O(n)$ space, whereas a dishonest user requires more than $n^2/20$ space to break security.

Guan and Zhandry [12] used the hardness of solving parity learning problem to construct key exchange, oblivious transfer, and bit commitment protocols in the bounded storage model.

Even after 28 years after the introduction of the bounded storage model, constructing a general multi-party computation protocol in this model using any techniques is still an open problem. To the best of our knowledge MPC in the bounded storage model has not yet been defined in any past work. In this work, we define semi-honest secure MPC in the bounded storage model and construct it based on the hardness of solving parity learning in bounded space.

1.1 Our Results and Technical Overview

In a multi-party computation scenario, there are k parties $\mathcal{P}_i$ ($i \in [k]$) each holding a private input x_i. The parties would like to know the output of a joint function $y = f(x_1, \cdots x_k)$ without leaking any information about its private input to the other parties. We would like to develop a protocol, where the parties

can send messages to each other in rounds and finally compute y based on the transcript. To enable the communication between parties, we model the parties as interactive Turing machines that have additional read and write communication tapes. Many MPC protocols have been proposed in the computational model [25,10,15,14] predominantly based on oblivious transfer (OT). As [12] showed how to build oblivious transfer in the bounded storage model based on the hardness of parity learning [20], one simple idea could be to directly substitute Guan Zhandry's OT construction in any of the existing MPC constructions.

However, there are few subtleties here. First, as we base our MPC protocol on the hardness of parity learning, we would like to show that if an adversary can break the security of our MPC protocol in bounded space, then he can solve the parity learning problem in bounded space. To model the adversarial behaviors in MPC properly, we model the adversaries as interactive Turing machines. However, the notion of bounded storage for interactive Turing machines is not well-defined in this context. We therefore need to first give a formal characterization of bounded-storage interactive Turing machines and show the parity learning hardness in the setting.

Review of Parity Learning for Branching Programs [20] We first give a brief overview on the branching program adversary model used in [20] and why we need to properly remodel the problem when it comes to Turing machines.

When proving the hardness of parity learning, [20] modeled the learning algorithm as a deterministic branching program and not as a probabilistic interactive Turing machine.

At a high level, a branching program is a graph, with vertices arranged in layers, and edges between the vertices in adjacent layers marked by parity learning samples $(a, b) \in \{0,1\}^n \times \{0,1\}$. Intuitively, each layer of the graph represents a time step and the vertices in each layer represent the possible states of the learning algorithm at that time step. The learner initializes his state with the vertex in the first layer. When the learner receives a stream of parity learning samples $(a_1, b_1), (a_2, b_2), \cdots$, the learner follows the edges corresponding to the samples. When it reaches a vertex in the last layer, it outputs a key $k \in \{0,1\}^n$ depending on the vertex. [20] showed that either the width of the branching program (number of vertices in a layer) has to be 2^{cn^2} or the length of the branching program (total number of layers which also represents the number of samples) has to be exponential in n in order to solve the parity learning problem with non-negligible advantage. This implies any learner with access to an only polynomial number of samples should have space at least cn^2 (for some constant c) to store the state. Intuitively, [20] defined the notion of the learner's space as the amount of storage required to store the state of the learner and did not include the amount of space required to store the transition function.

Parity Learning Hardness for Bounded-Storage Turing Machines. In the Turing machine model, we let the learner receive a stream of parity learn-

ing samples via its read communication tape[1] and has to write its output key on the output tape. The learner can additionally sample randomness via its random tape, interact with other Turing machines using its other communication tapes, and additionally take an advice string on its input tape.[2] What should be an analogous definition of space in case of (probabilistic interactive) Turing machines?[3] Should we include the space required to store the transition function? What about the space required to store the advice string on input tape, or the space required to store the output?

To address these questions, we first define a notion called "space characteristic" for Turing machines. We then prove that any Turing machine with bounded *space characteristic* has an exponentially small probability of solving the parity learning problem with a polynomial number of samples.

In order to prove the hardness result, we show that for every Turing machine that solves the parity learning problem, there exists a corresponding branching program that solves the problem with the same advantage. For simplicity, let us first consider the case of deterministic Turing machines which do not interact with other parties. Roughly speaking, we define a configuration of the Turing machine to be a tuple containing (state of the TM, input tape pointer, work tape contents, work tape pointer, output tape contents, output tape pointer). In this case, the vertices of the constructed branching program correspond to all possible configurations of the Turing machine, and the edges between the vertices correspond to how the Turing machine configuration changes when given a parity learning sample on its read comm. tape. We define the space characteristic of the TM to be the space required to store a configuration. Note that this does not include the storage required for input tape contents or state transition function. We give a more general definition of "space characteristic" for non-uniform probabilistic interactive TMs in Definition 1, and extend the hardness result to these general Turing machine learners in Section 3.

MPC in Bounded Storage Model. In this work, we construct a semi-honest secure k-party MPC protocol secure against $k-1$ corruptions in the bounded storage model. In this model, the corrupted parties try to learn more information from the union of their protocol transcripts but do not deviate from the protocol. As we base our security on the hardness of parity learning problem,

[1] The Turing machine has only read once access to this tape and thereby cannot move its tape head to the left.

[2] Looking forward, given an adversary that breaks the security of our MPC protocol, we build a reduction algorithm that solves the parity learning problem. In this case, the reduction algorithm interacts with the adversary. The reduction algorithm uses the circuit C and input tuple $(x_1, \cdots x_k)$ for which the adversary has a high advantage as an advice string written on its input tape.

[3] We note that [12] built various protocols where the parties are modeled as Turing machines. Unfortunately, they ignored the gap between Raz's theorem [20] and Turing machines, and used the traditional space complexity definition to define the space of a Turing machine.

the honest parties can run the protocol in $O(n)$ space, whereas the dishonest parties need at least $\Omega(n^2)$ space to break the protocol.

At a high level, we follow the GMW semi-honest MPC protocol approach [10]. The functionality is first represented as a circuit C containing only XOR and AND gates. When the circuit C is evaluated on input $(x_1, \cdots, x_k)$, let the bit value obtained at each wire w be v_w. The goal is to enable all the parties $\mathcal{P}_i$ to hold a secret share $r_{i,w}$ of v_w (i.e., $\sum_i r_{i,w} = v_w \bmod 2$) for each wire w. In order to do this, each party $\mathcal{P}_i$ first secret shares its private input x_i with all the other parties. Now, each party holds a secret share of v_w for all the input wires w.

The parties now proceed to process gate by gate in a logical order. If a gate is an XOR gate, the parties simply XOR the shares of their input wires locally to obtain a share of the output wire i.e., perform $r_{i,c} = r_{i,a} + r_{i,b} \bmod 2$, where a, b are input wires and c is the output wire of the gate. If the gate is an AND gate, then the parties would like to obtain secret shares of the bit $v_a * v_b = (\sum_i r_{i,a}) * (\sum_i r_{i,b}) = (\sum_i r_{i,a} * r_{i,b}) + \sum_{i<j}(r_{i,a} * r_{j,b} + r_{i,b} * r_{j,a})$. Each party $\mathcal{P}_i$ could locally compute $r_{i,a} * r_{i,b}$ term. To secret share the $(r_{i,a} * r_{j,b} + r_{i,b} * r_{j,a})$ term, the party $\mathcal{P}_i$ first samples a bit $\alpha \leftarrow \{0,1\}$, and sets $m_{(p,q)} = \alpha + p * r_{i,b} + q * r_{i,a}$ for each $p, q \in \{0,1\}$. The parties $\mathcal{P}_i$ and $\mathcal{P}_j$ now run an 1-out-of-4 oblivious transfer protocol[4], where $\mathcal{P}_i$ acts as the sender with input messages $(m_{(0,0)}, m_{(0,1)}, m_{(1,0)}, m_{(1,1)})$, and $\mathcal{P}_j$ acts as the receiver with $(r_{j,a}, r_{j,b})$ as its choice. The party $\mathcal{P}_j$ adds the OT output $\alpha + (r_{i,a} * r_{j,b} + r_{i,b} * r_{j,a})$ to its share of output wire $r_{j,c}$. The party $\mathcal{P}_i$ adds α to its share of the output wire $r_{i,c}$. After processing all the gates, each party sends their secret share corresponding to all the output wires to the other parties. Each party can compute the desired output by summing up the recieved secret shares.

1-out-of-4 Oblivious Transfer Protocol. In order for the above construction to work, we need a semi-honest secure 1-out-of-4 bit oblivious transfer protocol in the bounded storage model. However, [12] built only a 1-out-of-2 bit oblivious transfer based on the hardness of parity learning. In this work, we provide a generic way to transform any 1-out-of-2 OT to 1-out-of-4 OT in the bounded storage model. Let the sender's input messages be (m_0, m_1, m_2, m_3) and the receiver's choice be d. At a high level, the sender samples 3 uniformly random mask bits $r_{i,1}, r_{i,2}, r_{i,3}$ for each message m_i. For each pair $(i, j) \in [3] \times [3]$, the receiver chooses to obtain either a mask bit of m_i or a mask bit of m_j by performing a 1-out-of-2 OT with the sender. At the end, the sender masks each message m_i with the corresponding mask bits and sends $y_i = m_i + r_{i,1} + r_{i,2} + r_{i,3}$ (for each $i \in [3]$) to the receiver. Clearly, during the 1-out-of-2 OT invocations, if the receiver always chooses to obtain the mask bit corresponding to m_d, then the receiver can decrypt y_d and obtain m_d.

[4] In 1-out-of-t oblivious transfer protocol, the sender takes t messages $(m_1, \cdots m_t)$ as input. The receiver chooses an index c and obtains the message m_c, without letting the sender know about the choice c and without gaining any information about the other messages.

The above transformation is secure because the receiver cannot obtain information about other messages as for each $i \neq d$, the receiver does not know a mask bit corresponding to m_i. The sender cannot obtain any information about the choice d because the only messages he receives are part of the underlying 1-out-of-2 OT protocol. As the underlying 1-out-of-2 OT is secure, the sender cannot obtain any more information throughout the protocol. The above transformation can be extended to a general 1-out-of-k case by running 1-out-of-2 OT protocol $\binom{k}{2}$ times.

1.2 Related Works

A recent concurrent work [5] achieves simulation-based security for MPC based on the method from [8], but in the slightly different streaming BSM, compared to the traditional BSM used in [8]: the honest parties are less restricted and meanwhile the adversary is given less power. The protocols built in [12](and thus ours) can be viewed as similar to the streaming BSM, but the honest parties only use a single or very limited number of "long" rounds where they stream long messages to each other and therefore achieve better in terms of communication complexity.

2 Preliminaries

In this section, we first recall the definition of interactive turing machines and define a new parameter of ITMs called "space characteristic". We then define the notions of multiparty computation and oblivious transfer in the bounded storage model.

2.1 Interactive Turing Machines

An interactive turing machine (ITM) is a multi-tape turing machine. The functionality and access restrictions of each tape are described below.

- Space Parameter tape (read-only): This tape stores a value of the form 1^n. Here n is called the "space parameter" which determines the upper bound on the space that could be used by the turing machine. This is analogous to the security parameter typically used in the computational model.
- Input tape (read-only): The ITM receives its input on this tape.
- Output tape (write only): The ITM places its final output on this tape.
- Work tape (read & write): The ITM uses this for its internal storage during the computation.
- Random tape (read once): The ITM receives random bits on this tape.
- r read communication tapes (read once): These tapes are used to receive messages from other turing machines in an interactive protocol. Each tape is given a unique identifier in the set $[r]$.
- w write communication tapes (write-once): These tapes are used to send messages to other turing machines in an interactive protocol. Each tape is given a unique identifier in the set $[w]$.

For the tapes with read once (or write-once) access, the state machine can access each bit on the tape only once i.e., the head pointer of the turing machine is only allowed to move in the forward direction. In this paper, we assume that a uniformly random bit is sampled and placed on a cell of the random tape only at the time step at which the cell is accessed by the turing machine. In an interactive protocol, we can connect write tape i of an ITM $\mathcal{A}$ to a read tape j of a different ITM $\mathcal{B}$. In such a case, every bit written by $\mathcal{A}$ on it's i^{th} write tape is copied immediately to the first blank cell of ITM $\mathcal{B}$'s j^{th} read tape. At every time step, if an ITM has a non-blank cell at any of its read tape heads, then the ITM has to definitely read the cell and move its head to right.[5] For any turing machine $\mathcal{A}$, we denote $\mathcal{A}(1^n)$ to be the turing machine obtained by fixing the space parameter tape to 1^n.

We now define a new parameter for an ITM called "space characteristic". Looking ahead, we construct a multi-party computation protocol and show that no (computationally unbounded) adversary with small *space characteristic* can break the security of the protocol.

Definition 1 (Space Characteristic). *Consider any interactive turing machine M with r read comm. tapes, w write comm. tapes, alphabet size a, and Q states in its state machine. Suppose for every random string on the random tape, the execution of M on an input x (of length ip) and space parameter n uses at most T work tape cells, rnd random tape cells[6] and outputs a string of length at most op on its output tape. Then the space characteristic of M on the input x is given by*

$$(T + \mathsf{op} + 2(r + w)) \cdot \log_2 a + \log_2(n \cdot Q \cdot T \cdot \mathsf{ip} \cdot \mathsf{op} \cdot \mathsf{rnd}).$$

Intuitively, the first term $(T+\mathsf{op}+2(r+w))\cdot\log_2 a$ is the total amount of storage (in bits) required to store work tape, output tape and few symbols of each of the communication tapes. The second term intuitively represents the space required to store the current state, work tape pointer, input tape pointer, output tape pointer, space tape pointer and random tape pointer locations. The necessity for the second term would be more clear in Section 3. Note that the definition of space characteristic is different from that of space complexity. Space complexity typically does not include the space required to store the output, whereas the definition of space characteristic includes op term. Moreover, the definition of space characteristic includes $\log Q$ term, which is typically not included in the space complexity.

Definition 2 (s-space bounded ITM). *For any function $s : \mathbb{N} \to \mathbb{N}$, we say that an interactive turing machine M is s-space bounded for an input class*

[5] Note that a turing machine could have multiple read tapes and all the tapes could receive messages from other turing machines at the same time. The turing machine has to read all the read tape cells at once, but can only copy one symbol to its work tape at a timestep. We solve this by problem by allowing our work tape to use large alphabet size.

[6] In case the turing machine does not use randomness, we fix $\mathsf{rnd} = 1$.

$\{\mathcal{X}_n\}_{n\in\mathbb{N}}$, *if for all space parameters* $n \in \mathbb{N}$, *and inputs* $x \in \mathcal{X}_n$, *the turing machine* M *has a space characteristic* $s(n)$.

When the input class is clear from the context, we simply call an ITM to be s-space bounded.

2.2 Secure Multiparty Computation

In this section, we define Multiparty Computation protocols in the Bounded Storage Model. In this scenario, we have k parties $\mathcal{P}_i$ ($i \in [k]$) each holding a private input x_i. Each party $\mathcal{P}_i$ would like to know the output of a joint function $y_i = f_i(x_1, \cdots x_k)$ without leaking any information about its private input to the other parties. This is represented using a k-party functionality $\mathbf{f} = (f_1, f_2, \cdots f_k)$. To compute $(y_1, \cdots y_k)$, we would like to develop a protocol, where the parties can send messages to each other in rounds. In each round, the message sent by $\mathcal{P}_i$ to $\mathcal{P}_j$ (for any $i, j \in [k]$) depends on $\mathcal{P}_i$'s input, its random coins and all the messages it received in the previous rounds. At the end of the protocol, each party $\mathcal{P}_i$ computes $y_i = f_i(x_1, x_2, \cdots, x_k)$. As we are working in the bounded storage model, we are only interested in protocols where the algorithm used by each party $\mathcal{P}_i$ has small space characteristic. More formally,

Definition 3 (MPC protocol). *A k-party MPC protocol Π is described by ITMs $(\Pi_i)_{i\in[k]}$ and a simulator* Sim[7]. *The i^{th} ITM is used by party $\mathcal{P}_i$. Each ITM has k read communication tapes and k write communication tapes. The simulator has k write tapes. For every $j \neq i \in [k]$, the j^{th} write communication tape of Π_i is connected to the i^{th} read communication tape of Π_j i.e., this tape is used by the party $\mathcal{P}_i$ to send messages to the party $\mathcal{P}_j$. To initiate the protocol for a functionality $\mathbf{f}$, each party $\mathcal{P}_i$ first places $(\mathbf{f}, x_i)$ on the input tape of its ITM Π_i and then runs the turing machine.*

Definition 4 (s-Correctness). *For any function $s : \mathbb{N} \to \mathbb{N}$, we say that a k-party MPC protocol $\Pi = (\Pi_i)_{i\in[k]}$ s-correctly computes a class of k-party functionalities $\{\mathcal{F}_n\}_{n\in\mathbb{N}}$ if for every space parameter $n \in \mathbb{N}$, every functionality $\mathbf{f} = (f_1, \cdots, f_k) \in \mathcal{F}_n$, for every input vector $(x_1, \cdots, x_k)$ of the functionality, every set of random coins used by Π, when each party Π_i is run on input $(\mathbf{f}, x_i)$ and space parameter 1^n, the output tape of each Π_i at the end of the protocol is equal to $f_i(x_1, \cdots, x_k)$, and each ITM Π_i has a space characteristic at most $s(n)$.*

Semi-honest Security We now provide a security definition for MPC protocols against semi-honest adversaries in the bounded storage model. Intuitively, a semi-honest adversary can corrupt any subset of parties $\mathcal{I}$ before starting the protocol and then obtain their inputs $\{x_i\}_{i\in\mathcal{I}}$, random coins and transcript of the messages received by the parties. However, the adversary cannot force

[7] The simulator is used only in the security definition but not in the real protocol.

the corrupted parties to deviate from the protocol. Let the view of the adversary be all the information obtained by the adversary. The folklore definition says that an MPC protocol Π for some functionality $\mathbf{f}$ is semi-honest secure if there exists a simulator such that no semi-honest adversary can distinguish between the following two distributions: (1) view obtained by running the protocol Π on input $\{x_i\}_{i\in[k]}$, and (2) the output of the simulator on input $\mathbf{f}, \mathcal{I}, \{x_i\}_{i\in\mathcal{I}}, \{f_i(x_1, x_2, \cdots, x_k)\}_{i\in\mathcal{I}}$. Intuitively, if an MPC protocol satisfies such a security definition, it guarantees that the adversary cannot learn any more information by looking at the random coins and transcript of the messages other than what he can learn based on the inputs and outputs of the corrupted parties.

However, such a security defintion would not work in bounded storage model. This is because the adversary and the simulator can only have bounded space s, and the view generated by the protcol execution could be much larger than s. As a result, the simulator may not be able to generate the complete view on its output tape and send it to the adversary. Therefore, we model the adversary and the simulator as bounded space interactive turing machines that exchange stream of bits via their read and write communication tapes. Formally,

Definition 5 (Semi-honest Security). *For any functions $s_1 : \mathbb{N} \to \mathbb{N}$, $s_2 : \mathbb{N} \to \mathbb{N}$ and $\epsilon : \mathbb{N} \to [0, 1]$, we say that a k-party protocol $\Pi = \{\Pi_i\}_{i\in[k]}$ (s_1, s_2, ϵ)-securely computes a class of k-party functionalities $\{\mathcal{F}_n\}_{n\in\mathbb{N}}$ if*
(1) The simulator Sim *is s_1-space bounded, and*
(2) For every s_2-space bounded adversary $\mathcal{A}$ with k read tapes there exists an integer N_0 s.t. for every space parameter $n > N_0$, for any functionality $\mathbf{f} = (f_1, f_2, \cdots f_k) \in \mathcal{F}_n$, all input tuples $\mathbf{x} = (x_1, \cdots, x_k)$ belonging to the domain of $\mathbf{f}$, and for every subset $\mathcal{I} \subset [k]$, we have

$$|\Pr[\mathsf{GameSH}^{\mathcal{A}}_{1^n,\mathbf{f},\mathcal{I},\mathbf{x}}(0) = 1] - \Pr[\mathsf{GameSH}^{\mathcal{A}}_{1^n,\mathbf{f},\mathcal{I},\mathbf{x}}(1) = 1]| \leq \epsilon(n),$$

where GameSH *is described in Figure 1 and the probability is taken over the random coins used by the simulator, the adversary and the challenger.*

2.3 Oblivious Transfer

In this section, we define 1-out-of-k oblivious transfer in bounded storage model. In our scenario, we have 2 parties – a sender and a receiver. The sender takes as input k message bits $\{m_i\}_{i\in[k]}$ and the receiver takes as input a selector $c \in [k]$. The goal is to enable the receiver to obtain the message m_c. At the same time, we do not want the receiver to learn anything about the other messages, and we do not want the sender to learn anything about the selector c. To that end, we model the sender and the receiver as interactive turing machines $(\Pi_{\mathsf{sender}}, \Pi_{\mathsf{receiver}})$ each with 1 read and 1 write tape. The parties receive their input on input tape, send messages to each other using their read and write communication tapes, and finally write their output to the output tape. In the bounded storage model, we assume both the sender and the receiver have bounded space characteristic. We require that an honest sender and receiver can run the protocol using at most

$\mathsf{GameSH}^{\mathcal{A}}_{1^n,\mathbf{f},\mathcal{I},\mathbf{x}}(b)$

- If $b = 0$: For each $i \in I$, the challenger connects the i^{th} write tape of the simulator Sim to the i^{th} read tape of the adversary $\mathcal{A}$. The challenger then runs the simulator $\mathsf{Sim}(1^n, (\mathbf{f}, \mathcal{I}, (x_j)_{j\in\mathcal{I}}, (f_j(\mathbf{x}))_{j\in\mathcal{I}}))$ and the adversary $\mathcal{A}(1^n, (\mathbf{f}, \mathcal{I}, \mathbf{x}))$.
- Else if $b = 1$: The challenger runs the adversary $\mathcal{A}(1^n, (\mathbf{f}, \mathcal{I}, \mathbf{x}))$. It then places $(\mathsf{Input}, (\mathbf{f}, x_i))$ on the i^{th} read tape of $\mathcal{A}$ for each $i \in \mathcal{I}$. The challenger runs the real protocol Π on input $(\mathbf{f}, \mathbf{x})$ and space parameter n. At each step of the protocol Π, if any party $i \in I$,
  - reads a bit c on its random tape, then the challenger places (sample, c) on the i^{th} read tape of the adversary $\mathcal{A}$.
  - receives a bit c from j^{th} read tape ($j \neq i$), then the challenger places $(\mathsf{receive}, j, c)$ on the i^{th} read tape of the adversary $\mathcal{A}$.
- At the end of the protocol, the adversary $\mathcal{A}$ writes a guess bit b' on its output tape, which is considered as the output of the game.

Fig. 1: Security game for MPC against semi-honest adversary $\mathcal{A}$

space s_1, and any dishonest (computationally unbounded) sender and receiver with space less than s_2 cannot break the protocol. For the sake of security definition, we add two additional ITMs $\mathsf{SenderSim}$ and $\mathsf{ReceiverSim}$, each having 1 write tape, to the protocol description. We first define the correctness of an OT protocol.

Definition 6 (s-Correctness). *We say that a protocol $\Pi = (\Pi_{\mathsf{sender}}, \Pi_{\mathsf{receiver}})$ s-correctly performs a 1-out-of-k OT if for every input vector $(m_1, \cdots, m_k) \in \{0,1\}^k$, every set of random coins used by the ITMs, the output tape of Π_{sender} is empty and the output tape of Π_r is m_c at the end of the protocol, and both Π_{sender} and Π_{receiver} uses at most s space through out the protocol.*

Semi-Honest Security In this section, we define the notion of semi-honest security of oblivious transfer in the bounded storage model. In the semi-honest model, the parties run the protocol honestly but try to deduce more information than what is described by the functionality from the view of the protocol. The security definition is analogous to the general semi-honest security of MPC protocols for the functionality $f(\{m_i\}_{i\in[k]}, c) = (\phi, m_c)$, where ϕ denotes the empty string. To be more concrete, the semi-honest OT protocol has to 2 security requirements – security against a semi-honest sender, and security against a semi-honest receiver. When the sender is semi-honest, we require that the view of the sender (which constitutes its input, sampled randomness and set of all messages received) can be simulated by a space bounded ITM which takes as input $\{m_i\}_{i\in[k]}$. We require that any space-bounded sender (adversary) cannot distinguish between the real view and the simulated view. As we allow the size of the protocol's view to be more than the bound on the space of the turing machines, the simulator does not send the entire view at once. Rather, the simulator sends the view to the adversary as a stream of bits via a communication tape. The adversary has only read once access to this stream of bits. If the adversary

needs to access any of the bits multiple times, it can copy onto its (bounded space) work tape. Similarly, when the receiver is semi-honest, we require that the view of the receiver can be simulated by a space bounded ITM which takes as input (c, m_c). Formally,

Definition 7 (Security against Semi-honest Sender). *For any functions $s_1 : \mathbb{N} \to \mathbb{N}$, $s_2 : \mathbb{N} \to \mathbb{N}$ and $\epsilon : \mathbb{N} \to [0,1]$, we say that a 1-out-of-k oblivious transfer (OT) protocol $\Pi = (\Pi_{\mathsf{sender}}, \Pi_{\mathsf{receiver}}, \mathsf{SenderSim}, \mathsf{ReceiverSim})$ is (s_1, s_2, ϵ)-secure against a semi-honest sender if*
(1) The simulator SenderSim *has space characteristic at most s_1, and*
(2) For every s_2-space bounded adversary $\mathcal{A}$ with 1 read tape, there exists an integer N_0 s.t. for all space parameters $n > N_0$, for all message bit tuples $\{m_i\}_{i\in[k]}$ and for every selector bit $d \in [k]$, we have

$$|\Pr[\mathsf{GameSHSender}^{\mathcal{A}}_{1^n,\{m_i\}_{i\in[k]},d}(0) = 1] - \Pr[\mathsf{GameSHSender}^{\mathcal{A}}_{1^n,\{m_i\}_{i\in[k]},d}(1) = 1]| \leq \epsilon(n),$$

where the game GameSHSender *is described in Figure 2 and the probability is taken over the random coins used by the simulator, adversary and challenger.*

$\mathsf{GameSH}^{\mathcal{A}}_{1^n,\{m_i\}_{i\in[k]},d}(\beta)$

- If $\beta = 0$: The challenger connects the write communication tape of the simulator SenderSim to the read communication tape of the adversary $\mathcal{A}$. The challenger then runs the simulator $\mathsf{SenderSim}(1^n, \{m_i\}_{i\in[k]})$ and the adversary $\mathcal{A}(1^n, (\{m_i\}_{i\in[k]}, d))$.
- Else if $\beta = 1$: The challenger runs the adversary $\mathcal{A}(1^n, (\{m_i\}_{i\in[k]}, d))$. It then runs the protocol Π with sender's input $\{m_i\}_{i\in[k]}$ and receiver's input d. The challenger then places (input, d) on the read tape of the adversary. Whenever Π_{sender} reads a bit b on its random tape, the challenger places (sample, b) on the read tape of the adversary $\mathcal{A}$. Similarly, whenever Π_{sender} receives a bit b from Π_{receiver}, then the challenger places $(\mathsf{receive}, b)$ on the read tape of the adversary $\mathcal{A}$.
- At the end of the protocol, the adversary $\mathcal{A}$ writes β' on its output tape, which is considered as the output of the game.

Fig. 2: Security game for OT against Semi-honest Sender $\mathcal{A}$

Definition 8 (Security against Semi-honest Receiver). *For any functions $s_1 : \mathbb{N} \to \mathbb{N}$, $s_2 : \mathbb{N} \to \mathbb{N}$ and $\epsilon : \mathbb{N} \to [0,1]$, we say that a 1-out-of-k oblivious transfer (OT) protocol $\Pi = (\Pi_{\mathsf{sender}}, \Pi_{\mathsf{receiver}}, \mathsf{SenderSim}, \mathsf{ReceiverSim})$ is (s_1, s_2, ϵ)-secure against a semi-honest receiver if*
(1) The simulator ReceiverSim *has space characteristic at most s_1, and*
(2) For every s_2-space bounded adversary $\mathcal{A}$ with 1 read tape, there exists an integer N_0 s.t. for all space parameters $n > N_0$, for all message bit tuples $\{m_i\}_{i\in[k]}$ and for every selector bit $d \in [k]$, we have

$$|\Pr[\mathsf{GameSHReceiver}^{\mathcal{A}}_{1^n,\{m_i\}_{i\in[k]},d}(0) = 1] - \Pr[\mathsf{GameSHReceiver}^{\mathcal{A}}_{1^n,\{m_i\}_{i\in[k]},d}(1) = 1]| \leq \epsilon(n),$$

where GameSHReceiver *is described in Figure 3 and the probability is taken over the random coins used by the simulator, adversary and challenger.*

$\mathsf{GameSHReceiver}^{\mathcal{A}}_{1^n,\{m_i\}_{i\in[k]},d}(\beta)$

- If $\beta = 0$: The challenger connects the write communication tape of the simulator $\mathsf{ReceiverSim}$ to the read communication tape of the adversary $\mathcal{A}$. The challenger then runs the simulator $\mathsf{ReceiverSim}(1^n, (d, m_d))$ and the adversary $\mathcal{A}(1^n, (\{m_i\}_{i\in[k]}, d))$.
- Else if $\beta = 1$: The challenger runs the adversary $\mathcal{A}(1^n, (\{m_i\}_{i\in[k]}, d))$. It then runs the protocol Π with sender's input $\{m_i\}_{i\in[k]}$ and receiver's input d. The challenger then places (input, d) on the read tape of the adversary. Whenever Π_{receiver} reads a bit b on its random tape, the challenger places (sample, b) on the read tape of the adversary $\mathcal{A}$. Similarly, whenever Π_{receiver} receives a bit b from Π_{sender}, then the challenger places $(\mathsf{receive}, b)$ on the read tape of the adversary $\mathcal{A}$.
- At the end of the protocol, the adversary $\mathcal{A}$ writes β' on its output tape, which is considered as the output of the game.

Fig. 3: Security game for OT against Semi-honest Receiver $\mathcal{A}$

3 Time-Space Lower Bound for Parity Learning for Turing Machines

In this section, we recall the time-space lower bounds for the parity learning problem proved in [20] and adapt the theorems in the context of turing machines. In the parity learning problem, a secret string k is sampled uniformly at random from $\{0,1\}^n$. A learner has to compute the secret k when given a stream of samples $(a_1, b_1), (a_2, b_2), \cdots$, where each a_i is sampled uniformly at random from $\{0,1\}^n$, and b_i is the inner product of a_i and k mod 2. Roughly speaking, Raz [20] proved that any (computationally unbounded) learner that uses less than $n^2/20$ space requires either an exponential number of samples or has an exponentially small probably of outputting the correct answer.

To prove the theorem, Raz modeled the learning algorithm as a branching program. A branching program of length ℓ and width d is a directed acyclic graph with vertices arranged as $\ell + 1$ layers, each layer containing at most d vertices. Roughly speaking, each layer represents a time step and a vertex in each layer represents the state of the learning algorithm. Vertices with out-degree 0 are called leaf vertices. The first layer has only one vertex, representing the initial state of the learner. Every non-leaf vertex in the first ℓ layers has 2^{n+1} outgoing edges connected to the vertices in the next layer. Each of the outgoing edges is labelled with an $n + 1$-bit string (a, b), where $a \in \{0,1\}^n, b \in \{0,1\}$. Intuitively, an edge labelled by (a, b) represents how the learner modifies its state after processing the sample (a, b). Every leaf vertex is associated with a subspace $S \subseteq \{0,1\}^n$. Given a stream of samples $(a_1, b_1), (a_2, b_2), \cdots$, the learner

follows the computation path defined by the branching program and outputs the subspace S associated with the final vertex. We interpret the output as the learner guessing that the secret $k \in S$. Formally, [20] proves the following theorem.

Theorem 1 ([20]). *For any $c < 1/20$, there exists $\alpha > 0$, such that for any $m \leq 2^{\alpha n}$, and a branching program $\mathcal{A}$ of width at most 2^{cn^2} and length m, that takes a stream of samples (a_1, b_1), $(a_2, b_2), \cdots (a_m, b_m)$, where k, a_i are sampled uniformly from $\{0,1\}^n$ and $b_i = a_i \cdot k \mod 2$ for every i, outputs $\tilde{k} \in \{0,1\}^n$, then $\Pr[\tilde{k} = k] \leq O(2^{-\alpha n})$.*

The [21] theorem states the lower bound only for deterministic branching programs, whereas we need to model our adversary as a probabilistic interactive turing machine. Therefore, we now prove an analogous time-space lower bound for the parity learning problem modeling the learner as a probabilistic ITM.

Time-Space Lower Bound for Deterministic Turing Machines. For the sake of simplicity, we first prove lower bounds for deterministic ITMs without the random tape. We later reduce the lower bounds for probabilistic ITMs to the lower bounds for deterministic ITMs.

In this model, the learner has a space parameter tape storing 1^n, an input tape, an output tape, and one read comm. tape on which he receives a stream of samples $(a_1, b_1), (a_2, b_2), \cdots$. At the end of the stream, a special symbol $\#$ is given as input to denote the end. The learner has to halt by outputting an n-bit string k on its output tape and moving its read comm. tape pointer to the cell next to $\#$. Note that we do not input the stream of samples via the input tape because the ITM has read-only access to the input tape, and the input tape pointer is allowed to move in both directions. As we consider non-uniform learners, the ITM is allowed to have any advice string written on its input tape.

We will present the theorem statement and the proof as below.

Theorem 2. *For any $c < 1/20$, there exists $\alpha > 0$, such that for every non-uniform deterministic turing machine $\mathcal{A}$ with 1 read comm. tape and space characteristic s_c, where $s_c(n) = cn^2$, for every space parameter $n \in \mathbb{N}$, if $\mathcal{A}(1^n)$ is run with any advice string x of length at most $2^{cn^2/4}$ on its input tape, and a stream of samples (a_1, b_1), $(a_2, b_2), \cdots (a_m, b_m)$ on its read comm. tape, where $m \leq 2^{\alpha n}$, k, a_i are sampled uniformly from $\{0,1\}^n$ and $b_i = a_i \cdot k \mod 2$ for every i, then if $\mathcal{A}$ outputs $\tilde{k} \in \{0,1\}^n$, the success probability $\Pr[\tilde{k} = k] \leq O(2^{-\alpha n})$.*

Proof. For the sake of contradiction, suppose there exists a constant $c < 1/20$ s.t. for any $\alpha > 0$, there exists a non-uniform deterministic s_c-space bounded ITM $\mathcal{A}_\alpha$, a space parameter n and an advice string x s.t. $\mathcal{A}_\alpha(1^n, x)$ solves parity learning problem with $m \leq 2^{\alpha n}$ samples and success probability $\epsilon > O(2^{-\alpha n})$. Using this, we show a contradiction to Raz's lower bound. Specifically, for any such $\mathcal{A}_\alpha$, space parameter n and advice string x, we construct a corresponding branching program $\mathcal{B}$ that solves the parity learning problem with success probability more than $O(2^{-\alpha n})$.

At any time step during the execution of a turing machine, let us define its configuration to be a tuple containing (state, work tape content, work tape pointer, output tape content, output tape pointer, input tape pointer, space tape pointer). Note that we do not include input tape content as part of the configuration as it does not change during the execution. In the constructed branching program $\mathcal{B}$ vertices in every layer correspond to the possible configurations in the turing machine. As per our definition of space characteristic (Definition 1), the number of possible configurations of $\mathcal{A}$ is $2^{s_c(n)}$ and therefore the width of the branching program $\mathcal{B}$ is $2^{s_c(n)}$. We now describe how the edges are connected between adjacent layers. Consider any vertex v of the branching program and let its corresponding configuration be con. Consider any $n+1$-bit string (a, b), where $a \in \{0,1\}^n, b \in \{0,1\}$. We run the turing machine $\mathcal{A}(1^n)$ starting from this configuration con by placing x on its input tape, (a, b) on read comm. tape and placing pointer for read comm. tape at the starting cell of (a, b). When the read comm. tape pointer first reaches the cell next to b, let its configuration be con'. Note that there exists only one possible configuration con' as $\mathcal{A}$ is deterministic.[8] In $\mathcal{B}$, we place a directed edge from vertex v to the vertex w in the next layer which corresponds to the configuration con'. We now describe how to associate any vertex v in the final layer with an n-bit string k. Let the configuration corresponding to v be con. Run the turing machine $\mathcal{A}(1^n)$ from this configuration con by placing x on its input tape, # on read comm. tape and placing the pointer for read comm. tape at #. When $\mathcal{A}$ runs and places its read comm. tape pointer to the cell next to #, let the output written on its output tape be y. We associate the vertex v with the string y. Note that the branching program $\mathcal{B}$ has length m, width 2^{cn^2} and solves parity learning with success probability ϵ which is more than $O(2^{-\alpha n})$, thereby violating Raz's time-space lower bound.

Time-Space Lower Bound for Probabilistic Turing Machines. We now give an analogous time-space lower bound theorem for probabilistic ITMs.

Theorem 3. *For any $c < 1/20$, there exists $\alpha > 0$, such that for every non-uniform probabilistic turing machine $\mathcal{A}$ with 1 read comm. tape and space characteristic s_c, where $s_c(n) = cn^2$, for every space parameter $n \in \mathbb{N}$, if $\mathcal{A}(1^n)$ is run with any advice string x on its input tape, and a stream of samples (a_1, b_1), $(a_2, b_2), \cdots (a_m, b_m)$ on its read comm. tape, where $m \leq 2^{\alpha n}$, k, a_i are sampled uniformly from $\{0,1\}^n$ and $b_i = a_i \cdot k \mod 2$ for every i, then if outputs $\tilde{k} \in \{0,1\}^n$, the success probability $\Pr[\tilde{k} = k] \leq O(2^{-\alpha n})$.*

Proof. For the sake of contradiction, suppose there exists a constant $c < 1/20$ s.t. for any $\alpha > 0$, there exists a non-uniform probabilistic s_c-space bounded ITM $\mathcal{A}_\alpha$, a space parameter n_α and an advice string x_α s.t. $\mathcal{A}_\alpha(1^{n_\alpha}, x_\alpha)$ solves

[8] Observe that the transition function in the resulting branching program is deterministic even if the output tape contents are not included in the configuration definition. We include the output tape contents in the configuration so that every vertex in the final layer corresponds to an output key as defined by the turing machine.

parity learning problem with success probability $\epsilon > O(2^{-\alpha n_\alpha})$ using at most $m \leq 2^{\alpha n_\alpha}$ samples. Using this, we show a contradiction to Raz's lower bound. Specifically, for any $\alpha > 0$, we construct a non-uniform deterministic $s_{c'}$-space bounded ITM $\mathcal{B}_\alpha$ and an advice string x'_α that solves parity learning problem with success probability more than $O(2^{-\alpha n_\alpha})$.

For any $\alpha > 0$, let the success probability of $\mathcal{A}_\alpha(1^{n_\alpha}, x_\alpha)$ in solving the parity learning problem be ϵ, where the probability is taken over the random coins used to create samples (a_i, b_i) and the random coins used by $\mathcal{A}_\alpha$. Let rnd be the upper bound on the random coins used by $\mathcal{A}_\alpha$[9]. We know that there exists a bit $r_1 \in \{0,1\}$ s.t. $\mathcal{A}_\alpha(1^{n_\alpha}, x_\alpha)$ solves the parity learning problem with success probability at least ϵ, when the first cell of $\mathcal{A}_\alpha$'s random tape is fixed to be r_1. Extending this argument, we know that there exists a rnd-bit string r_α s.t. $\mathcal{A}_\alpha(1^{n_\alpha}, x_\alpha)$ solves the parity learning problem with success probability at least ϵ, when the entire random tape is fixed to r_α. For any $\alpha > 0$, let us now construct the ITM $\mathcal{B}_\alpha$ along with an advice string x'_α. At a high level, we let x'_α be equal to x_α concantenated with r_α i.e., the random coins on which $\mathcal{A}_\alpha$ has high success probability are given as part of advice string to $\mathcal{B}_\alpha$. $\mathcal{B}_\alpha(1^{n_\alpha}, x'_\alpha)$ emulates $\mathcal{A}_\alpha(1^{n_\alpha}, x_\alpha)$ with random coins hardcoded to r_α. Whenever $\mathcal{A}_\alpha$ reads a bit from its random tape, $\mathcal{B}_\alpha$ reads the corresponding bit from its advice string.

We now analyze the space characteristic of $\mathcal{B}_\alpha$. Clearly, $\mathcal{B}_\alpha$ uses the same number of work tape and output tape cells as $\mathcal{A}_\alpha$. The space characteristic of $\mathcal{A}_\alpha$ has $\log(|x_\alpha| \cdot \mathsf{rnd})$ term in it, whereas the space characteristic of $\mathcal{B}_\alpha$ has only $\log(|x'_\alpha|) = \log(|x_\alpha| + \mathsf{rnd})$ term in it. The number of states of $\mathcal{B}$ is only a small constant times the number of states in $\mathcal{A}$. Therefore, $\mathcal{B}_\alpha$ is s_c-space bounded ITM. Moreover, the success probability of $\mathcal{B}_\alpha$ is at least ϵ, and thereby breaks the time-space lower bound for deterministic ITMs.

Time-Space Lower Bound for Interactive Turing Machines. In the above theorems, we restricted the learner to receive only the stream of parity learning samples and not interact with any other ITMs. Looking forward, we construct an OT protocol based on parity learning and prove that if there exists an adversary $\mathcal{A}$ that breaks OT security, then there exists a reduction algorithm $\mathcal{B}$ that solves parity learning problem within low space characteristic. In the proof, $\mathcal{B}$ has to interact with $\mathcal{A}$ to solve the parity learning problem. In the standard model where the adversary is computationally bounded, this is not an issue because $\mathcal{B}$ can internally interact with $\mathcal{A}$ in poly time. However, in the bounded storage model, we need to be careful as we need to ensure $\mathcal{B}$ can internally run the conversation with $\mathcal{A}$ in bounded space. As similar scenario occurs in every proof where we need to construct a reduction algorithm, we present a general theorem which roughly states that if $\mathcal{B}$ solves a problem by interacting with $\mathcal{A}$, then there exists another TM $\mathcal{C}$ which can solve the problem without interacting with any other ITMs. At a high level, $\mathcal{C}$ runs both $\mathcal{A}$ and $\mathcal{B}$ internally, emulates their interaction using its work tape, and finally outputs whatever $\mathcal{B}$ outputs.

[9] Note that $\mathcal{A}_\alpha$ can use only at most $2^{s_c(n)}$ bits of randomness as it is s_c-space bounded

Formally, in this model, we have two ITMs – $\mathcal{A}$ and $\mathcal{B}$. $\mathcal{B}$ has t_1+1 read comm. tapes and t_2 write comm. tapes for some integers t_1, t_2. Similarly, $\mathcal{A}$ has t_2 read comm. tapes and t_1 write comm. tapes. $\mathcal{B}$ receives a stream of bits sampled from some distribution $\mathcal{D}$ from a challenger on its first read comm. tape, and uses the rest of its tapes to send and receive messages from $\mathcal{A}$. Both $\mathcal{A}, \mathcal{B}$ are allowed to be randomized and have any advice string on their input tapes. At the end of the execution, the string written by $\mathcal{B}$ on its output tape is considered to be the output of $(\mathcal{A}, \mathcal{B})$ pair. We now show that there exists an ITM $\mathcal{C}$ with only 1 read tape using which $\mathcal{C}$ receives a stream of bits sampled from the challenger, s.t. for any distribution of $\mathcal{D}$ used by the challenger, the output distribution of $\mathcal{C}$ is same as that of $(\mathcal{A}, \mathcal{B})$.

Theorem 4. *For any pair of ITMs $(\mathcal{A}, \mathcal{B})$ described above with space characteristic s_A and s_B respectively, there exists another ITM $\mathcal{C}$ with 1 read comm. tape and space characteristic $2(s_A + s_B) + \Delta$, where Δ is a constant that depends only number of tapes in $(\mathcal{A}, \mathcal{B})$, s.t. for any pair of advice strings x_A, x_B given to $\mathcal{A}, \mathcal{B}$ respectively, there exists an advice string x_C, s.t. for any space parameter n and any distribution $\mathcal{D}$ used by the challenger, the output distribution of $(\mathcal{A}, \mathcal{B})$ is same as the output distribution of $\mathcal{C}$.*

Proof. Let us first construct the ITM $\mathcal{C}$ along with its advice string x_C. For each symbol β present in the alphabet of $\mathcal{A}, \mathcal{B}$, we include the symbol β along with a fresh symbol $\underline{\beta}$ in the alphabet of $\mathcal{C}$. We also include a fresh symbol \$ which we call 'tape separator'. Intuitively, $\mathcal{C}$ internally runs both $\mathcal{A}, \mathcal{B}$ by maintaining many of the $\mathcal{A}, \mathcal{B}$'s tapes on its work tape. Specifically, the work tape of $\mathcal{C}$ is divided into many sections separated by the symbol \$. Each of the sections is used to simulate one of the following – work tape of $\mathcal{A}$, work tape of $\mathcal{B}$, output tape of $\mathcal{A}$, communication tapes between $\mathcal{A}$ and $\mathcal{B}$. Let the total number of communication tapes between $\mathcal{A}$ and $\mathcal{B}$ be t.[10] The work tape of $\mathcal{C}$ looks as follows.

$\mathcal{A}'$s work tape \$ $\mathcal{B}'$s work tape \$ $\mathcal{A}'$s output tape \$ comm. tape 1 \$ $\cdots$ \$ comm. tape t \$

The sections reserved for internal communication tapes store only last cell that is not yet read by the other turing machine. The input tape of $\mathcal{C}$ is also divided into 2 sections to simulate the input tapes of $\mathcal{A}$ and $\mathcal{B}$. So, the advice string x_C is given by $x_A||\$||x_B$, where $||$ denotes concatenation. The output tape of $\mathcal{C}$ is used to store the output tape of $\mathcal{B}$. The random tape of $\mathcal{C}$ is used to provide randomness required to run both $\mathcal{A}$ and $\mathcal{B}$ internally. At any time step, the fresh symbols with an underscore are used to mark the tape heads for each of these internal tapes.

At a high level, $\mathcal{C}$ first runs the transition function of $\mathcal{A}$ internally for one step and then runs the transition function of $\mathcal{B}$ internally for another step and repeats. $\mathcal{C}$ internally runs the transition function of $\mathcal{A}$ (similarly for $\mathcal{B}$) in multiple

[10] Note that $\mathcal{B}$ has 1 additional read communication tape to receive a stream of bits from the challenger.

steps – (1) $\mathcal{C}$ first scans the entire work tape and input tape. For each section of the tapes, $\mathcal{C}$ stores the symbol at the internal tape head (the symbols with an underscore) in its state. (2) $\mathcal{C}$ then runs the transition function of $\mathcal{A}$ and stores the set of actions to be performed (such as moving internal tapes heads to left or right) in its state. (3) Finally, $\mathcal{C}$ scans the entire work tape, input tape, and output tapes and performs the required actions. (4) $\mathcal{C}$ finally switches its state to perform the next transition for $\mathcal{B}$. In order to enable $\mathcal{C}$ to perform these actions, we design the state space of $\mathcal{C}$ in the following way. A state of $\mathcal{C}$ is a tuple consisting of

$$(\mathcal{A}'\text{s state},\ \mathcal{B}'\text{s state},\ \text{mode},\ \text{tape head contents/actions to be performed},\ \text{current section})$$

Here, the *mode* indicates, whether $\mathcal{C}$ is currently reading contents of internal tape heads, or performing a transition for $\mathcal{A}$ or $\mathcal{B}$. *Current section* indicates the internal section at which the tape head of $\mathcal{C}$ is currently located at.

Let us now analyze the space characteristic of ITM $\mathcal{C}$. Let the number of states, number of work tape cells, number of input tape cells, number of output tape cells, alphabet size and total number of communication tapes $\mathcal{A}$ be $Q_A, W_A, \mathsf{ip}_A, \mathsf{op}_A, \alpha_A, t$ Let the corresponding values for ITM $\mathcal{B}$ be $Q_B, W_B, \mathsf{ip}_B, \mathsf{op}_B$, $\alpha_B, t+1$ respectively. The number of work tape cells of $\mathcal{C}$ including the tape separators is given by $W_C = (W_A + W_B + \mathsf{op}_A + 2t + 3)$. For the sake of simplicity, let us assume both $\mathcal{A}, \mathcal{B}$ use alphabet of same size α.[11] The number of input tape, output tape cells, and alphabet size of $\mathcal{C}$ is given by $\mathsf{ip}_C = \mathsf{ip}_A + \mathsf{ip}_B + 1$, $\mathsf{op}_C = \mathsf{op}_B$, $\alpha_C = 2 * \alpha$ respectively[12]. The number of states in $\mathcal{C}$ is given by $Q_C = Q_A \cdot Q_B \cdot (2\alpha_C)^t \cdot (\log t) \cdot K$, for some constant K. $\mathcal{C}$ has only one read communication tape to receive inputs from the challenger. The amount of randomness uses by $\mathcal{C}$ is $\mathsf{rnd}_C = \mathsf{rnd}_A + \mathsf{rnd}_B$. Therefore, the space characteristic of $\mathcal{C}$ is given by

$$\begin{aligned}
&(W_C + \mathsf{op}_C + 3)\log_2 \alpha_C + \log_2(n \cdot Q_C \cdot W_C \cdot \mathsf{ip}_C \cdot \mathsf{op}_C \cdot \mathsf{rnd}_C) \\
&\leq 2[(W_A + \mathsf{op}_A + 2t)\log_2(\alpha) + \log_2(n \cdot Q_A \cdot W_A \cdot \mathsf{ip}_A \cdot \mathsf{op}_A \cdot \mathsf{rnd}_A)] \\
&\quad + 2[(W_B + \mathsf{op}_B + 2t + 2)\log_2(\alpha) + \log(n \cdot Q_B \cdot W_B \cdot \mathsf{ip}_B \cdot \mathsf{op}_B \cdot \mathsf{rnd}_B)] + \Delta \\
&= 2(s_A + s_B) + \Delta
\end{aligned}$$

for some constant Δ that depends only on t. ∎

[11] If $\mathcal{A}$ uses an alphabet of size α_A and $\mathcal{B}$ uses an alphabet of size α_B, then we can first convert them into turing machines $\mathcal{A}', \mathcal{B}'$ both using an alphabet of size two, and then apply the Theorem 4. Each symbol of $\mathcal{A}$ is represented using $\log \alpha_A$ symbols in $\mathcal{A}'$. $\mathcal{A}'$ internally runs the transition function of $\mathcal{A}$ in multiple stages. It reads a sequence of $\log \alpha_A$ bits from each tape and stores as part of the state, and then apply the transition function of $\mathcal{A}$ by writing $\log \alpha_A$ symbols to each tape.

[12] We could eliminate the need of introducing a new symbol for the tape separator by reusing the other fresh symbols introduced in $\mathcal{C}$.

3.1 Indistinguishability Parity Learning

The Guan-Zhandry 1-out-of-2 OT construction [12] is based on indistinguishability version of the Raz's parity learning space lower bound so that they can give indistinguishability security proofs for the construction. To extend the hardness result for the computational problem to the hardness result for the indistinguishability problem, they use the Goldreich-Levin algorithm, denoted as GL for the rest of the paper.

Theorem 5 (Goldreich-Levin Algorithm [9]). *Assume that there exists a function $f : \{0,1\}^n \to \{0,1\}$ s.t. for some unknown $x \in \{0,1\}^n$, we have*

$$\Pr_{\boldsymbol{r}\in\{0,1\}^n}[f(r) = \langle \boldsymbol{x}, \boldsymbol{r}\rangle] \geq \frac{1}{2} + \epsilon$$

Then there exists an algorithm that runs in time $O(n^2\epsilon^{-4}\log n)$, makes $O(n^2\epsilon^{-4}\log n)$ oracle queries to the function f, and outputs x with probability $\Omega(\epsilon^2)$.

Importantly, the GL algorithm also has $O(n)$ space characteristic so that we can use it during reduction.

We review the definition for the indistinguishability security game of parity learning, denoted as $\mathsf{PL}_{\mathcal{A},\delta}(n,\ell)$.

Definition 9 (Indistinguishability Parity Learning $\mathsf{PL}_{\mathcal{A},\delta}(n,\ell)$). *The challenger's input is $(1^n, 1^\ell)$*

1. *The challenger chooses a random $k \in \{0,1\}^n$.*
2. *For $i = 1, \cdots, \ell$:*
   - *The challenger writes $(\boldsymbol{a}_i, b_i)$ on the communication tape, where $\boldsymbol{a}_i \leftarrow \{0,1\}^n$ is uniformly random and $b_i = \boldsymbol{a}_i \cdot \boldsymbol{k}$.*
3. *The challenger writes $(\boldsymbol{a}_\ell, b_\ell)$ on the communication tape, where $\boldsymbol{a}_\ell \leftarrow \{0,1\}^n$ is uniformly random and chooses a random bit $\delta \in \{0,1\}$:*
   - *If $\delta = 0$, $b_\ell = \boldsymbol{a}_\ell \cdot \boldsymbol{k}$.*
   - *If $\delta = 1$, b_ℓ is a random bit.*
4. *Finally, $\mathcal{A}$ outputs a guess δ' for δ. $\mathcal{A}$'s advantage is defined as $(\Pr[\delta' = \delta] - 1/2)$.*

Next, we give a security proof in the ITM setting so that we can use the indistinguishability parity learning lower bound in later sections.

Theorem 6. *For any $c < 1/20$, there exists $\alpha > 0$ such that for all ITM adversary $\mathcal{A}$ with cn^2-space characteristic and that receives at most $2^{\alpha n}$ parity learning tuples, $\mathcal{A}$ has advantage $O(2^{-\alpha n/2})$ in $\mathsf{PL}_{\mathcal{A},\delta}(n, 2^{\alpha n})$, for all $n \in \mathbb{N}$.*

Proof. For the sake of contradiction, suppose there is some $c < 1/40$ such that for all $\alpha > 0$, there exists an ITM adversary $\mathcal{A}$ with cn^2-space characteristic and using at most $2^{\alpha n}$ parity learning tuples solves the above PL problem with advantage $\beta = \Omega(2^{-\alpha n/2})$ in $\mathsf{PL}_{\mathcal{A},\delta}(n, 2^{\alpha n})$. We show that there exists an ITM adversary $\mathcal{A}'$ with $(2cn^2 + O(n))$-space characteristic, which uses at most $2^{\alpha n}$

parity learning tuples and solves the parity learning problem with advantage $\beta' = \Omega(2^{-\alpha n})$, which would contradict Theorem 3. There exists some N_0, such that for all $n > N_0$, there exists some $c' < 1/20$ such that $c'n^2 \geq 2cn^2 + O(n)$.

As a first step, we show that if an adversary can distinguish between $(\mathbf{a}_\ell, b_\ell \leftarrow \{0,1\})$ and $(\mathbf{a}_\ell, b_\ell = \mathbf{a}_\ell \cdot \mathbf{k})$ with large probability, then we can have an algorithm that outputs $f(\mathbf{a}_\ell) = \mathbf{a}_\ell \cdot \mathbf{k}$ on input $\mathbf{a}_\ell$ with large probability.

Claim. If there exists some $c_1 < 1/20$, we have an ITM $\mathcal{A}$ with space characteristic $c_1 n^2$ that can distinguish between $(\mathbf{a}_\ell, b_\ell \leftarrow \{0,1\})$ and $(\mathbf{a}_\ell, b_\ell = \mathbf{a}_\ell \cdot \mathbf{k})$ with probability p, given $\{(\mathbf{a}_i, b_i = \mathbf{a}_1 \cdot \mathbf{k})\}_{i=1,\cdots,\ell-1}$ in a stream; then we can have an ITM $\mathcal{B}$ with space characteristic $c_2 n^2$ for some $c_2 < 1/20$, $\mathcal{B}$ outputs $\mathbf{a}_\ell \cdot \mathbf{k}$, with probability p.

Proof. Let $\mathcal{A}$ output (d, b_ℓ) where d is a bit indicating $\mathcal{A}$'s guess. If $\mathcal{A}$ thinks $b_\ell = \mathbf{a}_\ell \cdot \mathbf{k}$, then $d = 0$; if it thinks $b_\ell \leftarrow \{0,1\}$, then $d = 1$. $\mathcal{B}$ outputs $(b_\ell + d)$ as its guess for the value $\mathbf{a}_\ell \cdot \mathbf{k}$. Since $\mathcal{A}$ is correct with probability p, then $\mathcal{B}$ outputs the correct value $\mathbf{a}_\ell \cdot \mathbf{k}$ with probability p.

Let the challenger in $\mathsf{PL}_{\mathcal{A},\delta}(n,\ell)$ be an adversary $\mathcal{A}'$ in a modified parity learning game. a tuple in which b_ℓ can be a real inner product or uniformly random described as below.

Definition 10 (Modified Parity Learning $\mathsf{MPL}_{\mathcal{A},\delta}(n,m,\ell)$). *The challenger's input is* $(1^n, 1^m, 1^\ell)$.

1. *The challenger chooses a random* $k \in \{0,1\}^n$.
2. *For* $i = 1, \cdots, m$:
   If $i < \ell$:
   - *The challenger writes* $(\boldsymbol{a}_i, b_i)$ *on the communication tape, where* $\boldsymbol{a}_i \leftarrow \{0,1\}^n$ *is uniformly random and* $b_i = \boldsymbol{a}_i \cdot \boldsymbol{k}$.

   Else if $i \geq \ell$:
   - *The challenger writes* $(\boldsymbol{a}_i, b_i)$ *on the communication tape, where* $\boldsymbol{a}_i \leftarrow \{0,1\}^n$ *is uniformly random and chooses a random bit* $\lambda \leftarrow \{0,1\}$:
     - *If* $\lambda = 0$, $b_i = \boldsymbol{a}_i \cdot \boldsymbol{k}$.
     - *If* $\lambda = 1$, b_i *is a random bit.*
3. *Finally,* $\mathcal{A}$ *outputs a guess* $\boldsymbol{k}'$ *for* $\boldsymbol{k}$ *and wins if* $\boldsymbol{k}' = \boldsymbol{k}$.

$\mathcal{A}'$ receives parity learning tuples $(\mathbf{a}_i, b_i)_{i=1,\cdots,\ell}$ from the $\mathsf{MPL}_{\mathcal{A},\delta}(n,m,\ell)$ challenger and writes it on the communication tape to $\mathcal{A}$. The first $(\ell-1)$ tuples are used as the stream of parity learning tuples in $\mathsf{PL}_{\mathcal{A},\delta}(n,\ell)$ and the rest $(m-\ell)$ tuples each is used as a challenge tuple in $\mathsf{PL}_{\mathcal{A},\delta}(n,\ell)$.

The advantage of $\mathcal{A}$ is $\beta = |\Pr[\mathsf{PL}_{\mathcal{A},0}(n,\ell) = 0] - \Pr[\mathsf{PL}_{\mathcal{A},1}(n,\ell) = 1]|$. In other words, we have an ITM $\mathcal{A}$ that given $(\mathbf{a}_1, b_1), \cdots, (\mathbf{a}_{\ell-1}, b_{\ell-1})$ on the read tape, can distinguish between $(\mathbf{a}_\ell, b_\ell \leftarrow \{0,1\})$ and $(\mathbf{a}_\ell, b_\ell = \mathbf{a}_\ell \cdot \mathbf{k})$ with probability $(1+\beta)/2$. According to Section 3.1, given unknown $\mathbf{k} \in \{0,1\}^n$ and $\mathbf{a}_\ell \in \{0,1\}^n$ the function $f(\mathbf{a}_\ell) = \langle \mathbf{a}_\ell, \mathbf{k} \rangle$ can be computed with probability $(1+\beta)/2$.

After receiving every ℓ tuples from the $\mathsf{MPL}_{\mathcal{A},\delta}(n, m, \ell)$ challenger, $\mathcal{A}'$ can obtain $f(\mathbf{a}_i) = \langle \mathbf{a}_i, \mathbf{k} \rangle$ for each $\mathbf{a}_i$ with probability $(1 + \beta)/2$, where $i \geq l$, ; $\mathcal{A}'$ runs the GL algorithm from Theorem 5 along with obtaining $O(n^2\beta^4 \log n)$ number of inner products of $\mathbf{k}$ with $\mathbf{a}_i$ by invoking $\mathcal{A}$. By Theorem 5 $\mathcal{A}'$ can output correct $\mathbf{k}$ with probability $\Omega(\beta^2)$ in the end. $\mathcal{A}'$ simulates $\mathcal{A}$ using cn^2 space and running GL algorithm takes $O(n)$ space; $\mathcal{A}$'s space characteristic is $(2cn^2 + O(n))$; according to Theorem 3, $\mathcal{A}'$'s advantage is $O(2^{-\alpha n})$ and we must have $\Omega(\beta^2) \leq O(2^{-\alpha n})$. Hence, using cn^2 space and at most $2^{\alpha n}$ tuples, adversary $\mathcal{A}$'s advantage is at most $\beta = O(2^{-\alpha n/2})$.

4 1-out-of-4 Semi-Honest OT Construction

In this section, we describe our 1-out-of-4 oblivious transfer based on 1-out-of-2 oblivious transfer in the bounded storage model. In summary, we perform six 1-out-of-2 oblivious transfer with random input bits.

1. Let the sender's input be (m_0, m_1, m_2, m_3), and the receiver's input be $t \in \{0, 1, 2, 3\}$.
2. For each pair $(a, b) \in \{0, 1, 2, 3\}^2$ s.t $a < b$,
   Sender samples uniformly random bits denoted by $r^a_{a,b}, r^b_{a,b} \leftarrow \{0, 1\}$. If $t = a$, receiver sets $c = 0$, else if $t = b$ receiver sets $c = 1$, else receiver samples $c \leftarrow \{0, 1\}$. The sender and the receiver performs 1-out-of-2 OT with sender's input $(r^a_{a,b}, r^b_{a,b})$ and receiver's input c.
3. The Sender sends the following 4 bits (y_0, y_1, y_2, y_3) to the receiver.

$$y_0 = m_0 + r^0_{0,1} + r^0_{0,2} + r^0_{0,3}$$

$$y_1 = m_1 + r^1_{0,1} + r^1_{1,2} + r^1_{1,3}$$

$$y_2 = m_2 + r^2_{0,2} + r^2_{1,2} + r^2_{2,3}$$

$$y_3 = m_3 + r^3_{0,3} + r^3_{1,3} + r^3_{2,3}$$

4. The receiver computes m_t from y_t as he has all the 3 mask bits from the 1-out-of-2 oblivious transfers.

We will leave the proofs for correctness and security to Appendix A. We also review the 1-of-2 OT construction from [12] in Appendix B.

5 MPC Protocol in Bounded Storage

In this section, we describe our k-party semi-honest secure MPC protocol (for any integer k) in the bounded storage model from semi-honest secure 1-out-4 OT protocol in the bounded storage model.

Consider any functionality $\mathbf{f} = (f_1, f_2, \cdots f_k)$. Let the private input of each party $\mathcal{P}_i$ be x_i. The party $\mathcal{P}_i$ intends to compute $f_i(x_1, x_2, \cdots x_k)$. We first

represent the functionality as a circuit C containing only XOR and AND gates, which takes the inputs of all the parties and computes the output of all the parties. Let us denote the space parameter used by the protocol to be n, and let $\mathcal{C}_n$ be the set of all n-gate circuits. We describe our MPC protocol for the class of functionalities $\{\mathcal{C}_n\}_{n\in\mathbb{N}}$. We show that an honest party can run the protocol within $O(n)$ space characteristic, whereas any dishonest set of parties require at least $O(n^2)$ space characteristic to break the security.

Notations. Let us first introduce some notations. We order the gates of the circuit in such a way that $g_1, g_2 < g_3$ if the output wires of gates g_1, g_2 are used as input wires for gate g_3. We call such an order as a logical order. We denote the set of input wires and output wires of the circuit that belong to the party $\mathcal{P}_i$ be $\mathsf{Input}(i)$ and $\mathsf{Output}(i)$ respectively. Also for any input wire w, let the party that holds the private input corresponding to the wire be $\mathsf{Party}(w)$. Similarly, for any output wire w, let the party that is entitled to recieve output from the wire be $\mathsf{Party}(w)$. Let the input of the circuit C on input $\mathbf{x} = (x_1, \cdots, x_k)$ be $\mathbf{y} = (y_1, \cdots, y_k)$. For any input wire w that belongs to the party $\mathcal{P}_i$, let $\mathcal{P}_i$'s input bit corresponding to the wire w be $x_i[w]$. Similarly, for any output wire w that belongs to the party $\mathcal{P}_i$, let $\mathcal{P}_i$'s output bit corresponding to the w be $y_i[w]$. When the circuit C is run on the input $(x_1, \cdots, x_k)$, let the value at any wire w be v_w.

At a high level, the algorithm is similar to the GMW protocol [10]. Each party $\mathcal{P}_i$ first secret shares its input x_i with all the parties. Each of the parties now holds a secret share of v_w for each input wire w. The parties now run a 1-out-of-4 OT protocol to compute secret shares of v_w for each of the internal and output wires. Finally, for each output wire w, each party sends its secret share of v_w to the party that is entitled to receive the value of wire w (i.e., $\mathsf{Party}(w)$). Each party $\mathcal{P}_i$ now computes its output from the received shares.

Let $\mathsf{OT}_4 = (\Pi_{\mathsf{sender}}, \Pi_{\mathsf{receiver}}, \mathsf{SenderSim}, \mathsf{ReceiverSim})$ be a 1-out-of-4 oblivious transfer protocol secure in the bounded storage model. For the sake of simplicity, we assume that the reciever's choice in the OT_4 protocol is chosen from $\{(0,0),(0,1),(1,0),(1,1)\}$ intead of $\{0,1,2,3\}$. The algorithm used by each party $\mathcal{P}_i$ is formally described in Figure 4. Note that all the computations over bits are done over $\mathsf{GF}(2)$.

5.1 Security Proof for MPC Protocol

We present the proof of security and correctness for the MPC protocol in Section 5 here.

Correctness. Throughout the protocol, we maintain an invariant that the values $\{r_{i,w}\}_i$ for any wire w form a secret sharing of the bit v_w i.e., $\sum_{i\in[k]} r_{i,w} = v_w$. We can prove this via induction. Clearly, the invariant is satisfied if w is an input wire. Suppose the invariant is satisfied for the input wires a, b of any gate g. If g is an XOR gate, then $\sum_i r_{i,c} = \sum_i r_{i,a} + \sum_i r_{i,b} = v_a + v_b = v_c$ and the invariant is satisfied for the output wire. Suppose g is an AND gate, each pair of parties (i,j) perform 1-out-4 OT and obtain a secret sharing of $r_{i,a} * r_{j,b} + r_{i,b} * r_{j,a}$. Therefore,

$\Pi_i(1^n, C, x_i)$

Input Phase:

For each input wire $w \in \mathsf{Input}(i)$:
  For $j = 1$ to k s.t. $j \neq i$:
    Sample random bit $r_{j,w}$ and send it to party $\mathcal{P}_j$.
  Set $r_{i,w} = x_i[w] + \sum_{j \neq i} r_{j,w}$.
For each input wire $w \notin \mathsf{Input}(i)$:
  Receive bit $r_{i,w}$ from other parties. This corresponds to the party $\mathcal{P}_i$'s share of the input at wire w.

Eval Phase:
For each gate g in logical order:

- Let a, b be the input wires and c be the output wire of gate g.
- If g is XOR gate, set $r_{i,c} = r_{i,a} + r_{i,b}$.
- If g is AND gate:
  - For $j = 1$ to $i - 1$:
    Run $\mathsf{OT}_4.\Pi_{\mathsf{receiver}}$ algorithm with $(r_{i,a}, r_{i,b})$ as input and party $\mathcal{P}_j$ as the OT sender.
    Let the received message bit be β_j.
  - For $j = i + 1$ to k:
    Sample bit $\alpha_j \leftarrow \{0, 1\}$.
    For each $(x, y) \in \{0, 1\}^2$, compute message $m_{(x,y)} = \alpha_j + x * r_{i,b} + y * r_{i,a}$.
    Run $\mathsf{OT}_4.\Pi_{\mathsf{sender}}$ algorithm with input messages $(m_{(0,0)}, m_{(0,1)}, m_{(1,0)}, m_{(1,1)})$ and $\mathcal{P}_j$ as the OT receiver.
  - Set the party $\mathcal{P}_i$'s share for wire c as $r_{i,c} = r_{i,a} * r_{i,b} + \sum_{j=1}^{i-1} \beta_j + \sum_{j=i+1}^{k} \alpha_j$.

Output Phase:
For each output wire w:

- If $w \notin \mathsf{Output}(i)$, send the party $\mathcal{P}_i$'s share $r_{i,w}$ to the party $\mathsf{Party}(w)$.
- If $w \in \mathsf{Output}(i)$, receive secret shares of the wire w from the other parties i.e.,
  For $j = 1$ to k s.t. $j \neq i$:
    Receive bit $r_{j,w}$ from party $\mathcal{P}_j$.
  Compute the output bit $y[w] = \sum_{j=1}^{k} r_{j,w}$ and write it to the output tape.

Fig. 4: The algorithm used by the party $\mathcal{P}_i$ in the MPC protocol.

$\sum_i r_{i,c} = \sum_i r_{i,a} * r_{i,b} + \sum_{i \neq j}(r_{i,a} * r_{j,b} + r_{i,b} * r_{j,a}) = (\sum_i r_{i,a}) * (\sum_i r_{i,b}) = v_a * v_b = v_c$.

We now argue that if OT_4 is s-correct for some function $s : \mathbb{N} \to \mathbb{N}$, then all the parties in the above protocol have space characteristic at most $O(s + n)$ for the class of n-gate functionalities. Observe that work tape of each party needs to store at most 1 bit per each wire, along with work tape contents needed for oblivious transfer. Both the input and output tapes need to store at most n bits, as we are dealing with n-gate functionalities. Moreover, the number of states required by the protocol is only constant times that of the number of states in OT_4. Therefore, the above MPC protocol has space characteristic $O(s + n)$. If OT_4 scheme presented in the Section 4 is used, the space characteristic of the protocol is $O(n)$.

5.2 Proof of Security

We now prove that the above scheme is semi-honest secure against adversaries that have space characteristic less than $n^2/80$. Formally, we prove the following theorem.

Theorem 7. *For any functions $s_1, s_2, s_3 : \mathbb{N} \to \mathbb{N}$, and $\epsilon : \mathbb{N} \to [0, 1]$, assuming OT_4 is s_1-correct (as per Definition 6), (s_2, s_3, ϵ)-secure against semi-honest sender and (s_2, s_3, ϵ)-secure against semi-honest receiver (as per Definitions 7 and 8),*[13] *the above k-party MPC protocol is $(O(s_1 + s_2 + kn), s_2/2 - (s_1 + s_3 + O(n)), \epsilon \cdot n \cdot k^2)$ semi-honest secure for the class of the n-gate functionalities $\{\mathcal{C}_n\}_{n \in \mathbb{N}}$.*

Proof. To prove security, we first build a simulator Sim which given a circuit $\mathcal{C}$, a set of parties $\mathcal{I}$ along with their input-output pairs, outputs a view of the parties in the set $\mathcal{I}$. We then argue that the simulator has $O(s_1 + s_2 + kn)$ space characteristic, and prove that any adversary with space characteristic at most $1/2 \cdot s_3 - (s_1 + s_2 + O(n))$ cannot distinguish between the views generated by the simulator, and the views generated during the real execution of the protocol with advantage more than $\epsilon \cdot n \cdot k^2$ probability.

At a high level, the simulator works as follows. The simulator has k write communication tapes, each indexed by an integer in $[k]$. The simulator writes the view of the i^{th} party to its i^{th} communication tape, which we hereby denote $\mathsf{WriteTape}_i$. We present a formal description of the algorithm used by the simulator in Figure 5. Observe that the simulator has to internally run an OT_4 protocol, run $\mathsf{ReceiverSim}$ and $\mathsf{SenderSim}$ algorithms and store at most k bits per each gate. Therefore, the Sim can be run within space characteristic $O(s_1 + s_2 + kn)$.

We now prove that the distribution of the views generated by the real protocol and the simulator are indistinguishable for any space bounded adversary via a hybrid argument. The hybrids are indexed by a tuple (t, ℓ, m), where t is an

[13] For the sake of simplicity, we assume OT_4 is (s_2, s_3, ϵ)-secure against semi-honest receiver. A similar proof works even if we assume OT_4 is (s_4, s_5, ϵ') for some other functions s_4, s_5, ϵ'.

$\mathsf{Sim}(1^n, C, \mathcal{I}, (x_j)_{j\in\mathcal{I}}, (y_j)_{j\in\mathcal{I}})$

Input Phase:

For each party $i \in \mathcal{I}$:
  Include the input in the party P_i's view i.e., Write $(\mathsf{input}, (C, x_i))$ to $\mathsf{WriteTape}_i$.
For each party $i \in \mathcal{I}$ and each input wire $w \in \mathsf{Input}(i)$:
  For $j = 1$ to k s.t. $j \neq i$:
    Sample party $\mathcal{P}_j$'s share of the input at wire w as $r_{j,w} \leftarrow \{0,1\}$. Write $(\mathsf{sample}, r_{j,w})$ to $\mathsf{WriteTape}_i$. Set the party $\mathcal{P}_i$'s share of wire w to be $r_{i,w} = x_i[w] + \sum_{j\neq i} r_{j,w}$.
For each $i \in \mathcal{I}$ and each input wire $w \notin \mathsf{Input}(i)$:
  If $\mathsf{Party}(w) \notin \mathcal{I}$, sample bit $r_{i,w} \leftarrow \{0,1\}$.
  Write $(\mathsf{receive}, \mathsf{Party}(w), r_{i,w})$ to $\mathsf{WriteTape}_i$.

Eval Phase:
For each gate g in the logical order:

- Let a, b be the input wires and c be the output wire of gate g.
- If g is XOR gate, for each $i \in \mathcal{I}$, set $r_{i,c} = r_{i,a} + r_{i,b}$.
- If g is AND gate:
  - For each $i \in \mathcal{I}$, set $r_{i,c} = r_{i,a} * r_{i,b}$.
  - For each $i = 1$ to $k-1$ & $j = i+1$ to k:
    - If both $\mathcal{P}_i$ and $\mathcal{P}_j$ are in the set $\mathcal{I}$ ($i \in \mathcal{I}$ & $j \in \mathcal{I}$),
      - Sample bit $\alpha \leftarrow \{0,1\}$.
      - For each $(x, y) \in \{0,1\}^2$, compute the message bit $m_{(x,y)} = \alpha + x * r_{i,b} + y * r_{i,a}$.
      - Perform 1-out-of-4 OT internally with $(m_{(0,0)}, m_{(0,1)}, m_{(1,0)}, m_{(1,1)})$ as sender's input and $(r_{j,a}, r_{j,b})$ as receiver's input.
      - During the OT protocol, write the sender's transcript on tape $\mathsf{WriteTape}_i$ and receiver's transcript on $\mathsf{WriteTape}_j$.
      - Set $r_{i,c} = r_{i,c} + \alpha$ and $r_{j,c} = r_{j,c} + \alpha + r_{i,a} * r_{j,b} + r_{i,b} * r_{j,a}$.
    - Else if only $\mathcal{P}_i$ is in the set $\mathcal{I}$ ($i \in \mathcal{I}$ & $j \notin \mathcal{I}$),
      - Sample bit $\alpha \leftarrow \{0,1\}$ and set $r_{i,c} = r_{i,c} + \alpha$.
      - For each $(x, y) \in \{0,1\}^2$, compute the message bit $m_{(x,y)} = \alpha + x * r_{i,b} + y * r_{i,a}$.
      - Internally, run $\mathsf{SenderSim}(1^n)$ on input $(m_{(0,0)}, m_{(0,1)}, m_{(1,0)}, m_{(1,1)})$ using $\mathsf{WriteTape}_i$ as its write comm. tape.
    - Else if only $\mathcal{P}_j$ is in the set $\mathcal{I}$ ($i \notin \mathcal{I}$ & $j \in \mathcal{I}$),
      - Sample bit $\beta \leftarrow \{0,1\}$ and set $r_{j,c} = r_{j,c} + \beta$.
      - Internally, run $\mathsf{ReceiverSim}(1^n)$ on input $(\beta, (r_{j,a}, r_{j,b}))$ using $\mathsf{WriteTape}_i$ as its write comm. tape.

Output Phase:
For each output wire w s.t. $\mathsf{Party}(w) \in \mathcal{I}$,
  For each $i \notin \mathcal{I}$, sample $r_{i,w} \leftarrow \{0,1\}$ under the constraint $\sum_{i=1}^{k} r_{i,w} = y[w]$.
  For each $j \neq i$, write $(\mathsf{receive}, j, r_{j,w})$ on tape $\mathsf{WriteTape}_i$.

Fig. 5: Algorithm used by simulator to generate the view of parties in set $\mathcal{I}$. The simulator writes the view of party $\mathcal{P}_i$ on $\mathsf{WriteTape}_i$.

integer in $[1, n]$, and ℓ, m are integers in $[k]$ s.t. $\ell < m$. In any hybrid $H_{t,\ell,m}$, the hybrid simulator $\mathsf{HSim}_{t,\ell,m}$ (described in Figure 6) is used to generate a view of the parties in the set $\mathcal{I}$. Unlike the simulator Sim, this hybrid simulator is given access to inputs x_j of all the k parties.

At a high level, the hybrid simulator $\mathsf{HSim}_{t,\ell,m}$ algorithm first secret shares the input during the *input phase* just like the simulator Sim. The hybrid simulator, therefore, knows the secret shares $r_{i,w}$ for all the parties $\mathcal{P}_i$ in the set $\mathcal{I}$ and all the input wires w. Then during the *eval phase*, the hybrid simulator $\mathsf{HSim}_{t,\ell,m}$ processes all the gates $g < t$ just like the simulator Sim, and processes gates $g > t$ by internally running the real protocol. For the gate $g = t$, for all party pairs (i, j) s.t. $(i, j) \succeq (\ell, m)$[14], the hybrid simulator internally run a real oblivious transfer protocol, whereas for the party pairs $(i, j) \prec (\ell, m)$[15], the hybrid simulator simulates the view of the oblivious transfer. We formally describe the sequence of hybrids described in Figure 6. The differences from the Sim algorithm are marked in red color.

Throughout the execution, the hybrid simulator maintains an invariant that it knows secret share $r_{i,w}$ for any party $i \in \mathcal{I}$ and wire w. For any gate $g < t$ and its output wire c, the hybrid simulator $\mathsf{HSim}_{t,\ell,m}$ has no information about shares $r_{i,c}$ for parties $\mathcal{P}_i \notin \mathcal{I}$ after processing the gate g. In case the wire c is the output wire of the circuit or if c is used as input wire to another gate g' for which the real OT protocol is used to process the gate, then $r_{i,c}$ is sampled at that point. When processing the gate $g = t$, suppose the oblivious transfer for party pair (i, j) is simulated and the party $\mathcal{P}_i$ is not in the set $\mathcal{I}$. Then the party $\mathcal{P}_i$'s share $r_{i,c}$ of the output wire c that is computed by $\mathsf{HSim}_{t,\ell,m}$ is not complete, as the party $\mathcal{P}_i$'s output of the OT between (i, j) parties is not included in the share $r_{i,c}$. Therefore, $\mathsf{HSim}_{t,\ell,m}$ deletes the share $r_{i,c}$ in such a case. The hybrid simulator acts similarly if the oblivious transfer for party pair (i, j) is simulated and $\mathcal{P}_j$ is not the set $\mathcal{I}$.

Let us introduce some notations. For any set of integers $(g_1, i_1, j_1, g_2, i_2, j_2)$, we say that $(g_1, i_1, j_1) \prec (g_2, i_2, j_2)$ if $g_1 < g_2$ or $(g_1 = g_2 \wedge i_1 < i_2)$ or $(g_1 = g_2 \wedge i_1 = i_2 \wedge j_1 < j_2)$. We say that $(g_1, i_1, j_1) \succeq (g_2, i_2, j_2)$ if $(g_2, i_2, j_2) \prec (g_1, i_1, j_1) \vee (g_2, i_2, j_2) = (g_1, i_1, j_1)$. For any tuple of integers $(g, i, j) \in [n] \times [k] \times [k]$ s.t. $j > i$, we define the function $\mathsf{Next}(g, i, j) = \begin{cases} (g+1, 1, 2) & \text{if } i = k-1 \ \wedge \ j = k \\ (g, i+1, i+2) & \text{if } i < k-1 \ \wedge \ j = k \\ (g, i, j+1) & \text{if } j < k \end{cases}$.

Consider any adversary $\mathcal{A}$ which has k read tapes. Let $\Pr[H^{\mathcal{A}}_{t.\ell.m}(1^n, C, \mathcal{I}, (x_i)_{i\in[k]}) = 1]$ be the probability that $\mathcal{A}$ outputs 1 when its read tapes are connected to the write tapes of $\mathsf{HSim}(1^n, (C, \mathcal{I}, (x_i)_{i\in[k]}))$. We now prove that each adjacent pair of hybrids are indistinguishable to a space bounded adversary.

Lemma 1. *For any space parameter $n \in \mathbb{N}$, any circuit $C \in \mathcal{C}_n$, any input tuple $(x_1, \cdots, x_k)$ belonging to the domain of $\mathcal{C}$ and any set $\mathcal{I} \subset [k]$, the distribution*

[14] We say that $(i, j) \succeq (\ell, m)$ iff $(i > \ell) \vee (i = \ell \wedge j >= m)$.
[15] We say that $(i, j) \prec (\ell, m)$ if $(i < \ell) \vee (i = \ell \wedge j < m)$.

The algorithm used by $\mathsf{HSim}_{t,\ell,m}(\mathcal{I}, C, (x_j)_{j\in[k]})$:

Input Phase: The input phase runs similar to the Sim algorithm. Additionally, for each input wire w, set $v_w = \mathbf{x}[w]$.

Eval Phase:

For each gate g in the logical order:

- Let a, b be the input wires and c be the output wire of gate g.
- If g is XOR gate, set $v_c = v_a + v_b$. For each $i \in \mathcal{I}$, $r_{i,c} = r_{i,a} + r_{i,b}$.
- If g is AND gate, set $v_c = v_a * v_b$.
  - For each $i \in \mathcal{I}$, set $r_{i,c} = r_{i,a} * r_{i,b}$.
  - For each $i = 1$ to $k-1$ & $j = i+1$ to k:
    - If $(g, i, j) \succeq (t, \ell, m)$ and one of the parties i, j is in $\mathcal{I}$:
      - For any party $p \in \{i, j\}$ and wire $w \in \{a, b\}$, if bit $r_{p,w}$ is not set, then sample $r_{p,w}$ uniformly at random subject to constraint $\sum_{q=1}^{k} r_{q,w} = v_w$, and set $r_{p,c} = r_{p,a} * r_{p,b}$.
      - Sample bit $\alpha \leftarrow \{0,1\}$.
      - For each $(x, y) \in \{0,1\}^2$, compute the message bit $m_{(x,y)} = \alpha + x * r_{i,b} + y * r_{i,a}$.
      - Perform 1-out-of-4 OT internally with $(m_{(0,0)}, m_{(0,1)}, m_{(1,0)}, m_{(1,1)})$ as sender's input and $(r_{j,a}, r_{j,b})$ as receiver's input.
      - During the OT protocol, write the sender's transcript on tape $\mathsf{WriteTape}_i$ and receiver's transcript on $\mathsf{WriteTape}_j$.
      - Set $r_{i,c} = r_{i,c} + \alpha$ and $r_{j,c} = r_{j,c} + \alpha + r_{i,a} * r_{j,b} + r_{i,b} * r_{j,a}$.
    - Else if both $\mathcal{P}_i$ and $\mathcal{P}_j$ are in the set $\mathcal{I}$ ($i \in \mathcal{I}$ & $j \in \mathcal{I}$), run similar to the Sim algorithm
    - Else if only $\mathcal{P}_i$ is in set $\mathcal{I}$ ($i \in \mathcal{I}$ & $j \notin \mathcal{I}$), run similar to the Sim algorithm
    - Else if only $\mathcal{P}_j$ is in set $\mathcal{I}$ ($i \notin \mathcal{I}$ & $j \in \mathcal{I}$), run similar to the Sim algorithm.
  - If there exists a party $\mathcal{P}_i$ not in the set $\mathcal{I}$, and some other party $\mathcal{P}_j$ s.t. either $(i, j) \prec (\ell, m)$ or $(j, i) \prec (\ell, m)$, then delete the bit $r_{i,c}$.

Output Phase:

For each output wire w s.t. $\mathsf{Party}(w) \in \mathcal{I}$,

For each $i \notin \mathcal{I}$ s.t. $r_{i,w}$ is not set, sample $r_{i,w} \leftarrow \{0,1\}$ under the constraint $\sum_{i=1}^{k} r_{i,w} = v_w$.

For each $j \neq i$, write $(\mathsf{receive}, j, r_{j,w})$ on tape $\mathsf{WriteTape}_i$.

Fig. 6: Algorithm used by the hybrid simulator $\mathsf{HSim}_{t,\ell,m}$ ($t \in [n], \ell \in [k], m > \ell$) to generate the view of set of parties $\mathcal{I}$. The simulator writes the view of party $\mathcal{P}_i$ on $\mathsf{WriteTape}_i$.

of views of parties in the set $\mathcal{I}$ generated by the real protocol is identical to the views generated by $\mathsf{HSim}_{1,1,2}$.

Proof. This follows from the definition of the hybrids.

Lemma 2. *For any functions $s_1, s_2, s_3 : \mathbb{N} \to \mathbb{N}$ and $\epsilon : \mathbb{N} \to [0,1]$, assuming OT_4 is s_1-correct (as per Definition 6), (s_2, s_3, ϵ)-secure against semi-honest sender and (s_2, s_3, ϵ)-secure against semi-honest receiver (as per Definitions 7 and 8), then for every $s_3/2 - (s_1 + s_2 + kn)$-space bounded adversary $\mathcal{A}$, there exists an integer N_0 s.t. for every space parameter $n > N_0$, for every circuit $C \in \mathcal{C}_n$, every set of parties $\mathcal{I} \subset [k]$, every input tuple $(x_1, \cdots, x_k)$ belonging to the circuit's domain, any indices $(t, \ell, m) \in [n] \times [k] \times [k]$ s.t. $\ell < m$, the advantage $|\Pr[H^{\mathcal{A}}_{t.\ell.m}(1^n, C, \mathcal{I}, (x_i)_{i\in[k]}) = 1] - \Pr[H^{\mathcal{A}}_{t'.\ell'.m'}(1^n, C, \mathcal{I}, (x_i)_{i\in[k]}) = 1]| \leq \epsilon(n)$, where $(t', \ell', m') = \mathsf{Next}(t, \ell, m)$.*

Proof. For the sake of contradiction, let us assume there exists an $s_2/2 - (s_1 + s_3 + kn)$-space bounded adversary $\mathcal{A}$ s.t. for every integer N_0, there exists an $n > N_0$, circuit $C \in \mathcal{C}_n$, set of parties $\mathcal{I} \subset [k]$, input tuple $(x_1, \cdots, x_k)$, indices $(t, \ell, m) \in [n] \times [k] \times [k]$ s.t. $\ell < m$, the advantage $|\Pr[H^{\mathcal{A}}_{t.\ell.m}(1^n, C, \mathcal{I}, (x_i)_{i\in[k]}) = 1] - \Pr[H^{\mathcal{A}}_{t'.\ell'.m'}(1^n, C, \mathcal{I}, (x_i)_{i\in[k]}) = 1]| > \epsilon(n)$, where $(t', \ell', m') = \mathsf{Next}(t, \ell, m)$. We now show how to break the semi-honest security of the underlying OT_4 scheme.

We know that the Hybrids $H_{t.\ell.m}$ and $H_{t'.\ell'.m'}$ are identical if (1) the gate t is an XOR gate or (2) t is an AND gate and both the parties $\mathcal{P}_\ell, \mathcal{P}_m$ are in the set $\mathcal{I}$ or (3) t is an AND gate and both $\mathcal{P}_\ell, \mathcal{P}_m$ are not in the set $\mathcal{I}$. As the adversary $\mathcal{A}$ can distinguish between these hybrids, we know that t is an AND gate and exactly one of the parties $\mathcal{P}_\ell, \mathcal{P}_m$ is not in the set $\mathcal{I}$. Suppose $\mathcal{P}_\ell \notin \mathcal{I}$ and $\mathcal{P}_m \in \mathcal{I}$, we break the security of OT_4 scheme against semi-honest receiver[16]. Specifically, we show that there exists an s_3-space bound adversary $\mathcal{A}'$ s.t. for every N_0', there exists a space parameter $n' > N_0'$, message bits $(m_{(0,0)}, m_{(0,1)}, m_{(1,0)}, m_{(1,1)})$ and receiver's choice (d_1, d_2) s.t. the advantage in $\mathsf{GameSHReceiver}$ is atleast $\epsilon(n')$.

We prove the argument in two steps – We first show that there exists a pair of ITMs $(\mathcal{A}_1, \mathcal{A}_2)$ s.t. $\mathcal{A}_1$ is $1/2 \cdot s_3 - (s_1 + s_2 + kn)$-space bounded, $\mathcal{A}_2$ is $(kn + s_1 + s_2)$-space bounded, $\mathcal{A}_2$ receives oblivious transfer challenge view on one of its read tapes, interacts with $\mathcal{A}_1$ using its other tapes and finally solves the oblivious transfer game $\mathsf{GameSHReceiver}$ with ϵ advantage. We then invoke Theorem 4 to construct an s_3-space bounded adversary $\mathcal{A}'$ that solves game $\mathsf{GameSHReceiver}$ with ϵ advantage. At a high level, we simply set $\mathcal{A}_1$ to be equal to the adversary $\mathcal{A}$ which can distinguish between the hybrids (t, ℓ, m) and (t', ℓ', m') with advantage more than ϵ. The algorithm $\mathcal{A}_2$ takes $(C, \mathcal{I}, (x_1, \cdots, x_k), t, \ell, m)$ on its input tape as an advice string. The algorithm $\mathcal{A}_2$ has k write tapes that are connected to the k read tapes of $\mathcal{A}_1$. $\mathcal{A}_2$ runs similar to the $\mathsf{HSim}_{t.\ell.m}$ algorithm to generate the views for each oblivious transfer corresponding to the pairs $(g, i, j) \neq (t, \ell, m)$. When processing the gate t and oblivious transfer for party pair (ℓ, m), $\mathcal{A}_2$

[16] In case, $\mathcal{P}_\ell \in \mathcal{I}$ and $\mathcal{P}_m \notin \mathcal{I}$, we can break the security of OT_4 scheme against semi-honest sender. This case can be handled analogously.

samples a random bit $\alpha \leftarrow \{0,1\}$ and invokes OT challenger with the sender's input sampled similar to $\mathsf{HSim}_{t.\ell.m}$, and the receiver's input as $(\alpha, r_{j,a}, r_{j,b})$. The OT challenger samples a bit $b \leftarrow \{0,1\}$. If $\beta = 0$, the OT challenger sends the real view of party $\mathcal{P}_j$ in the oblivious transfer to $\mathcal{A}_2$. If $\beta = 1$, the OT challenger sends the simulated view of party $\mathcal{P}_j$ in the oblivious transfer to $\mathcal{A}_2$. $\mathcal{A}_2$ copies this message onto its j^{th} write tape i.e., includes party $\mathcal{P}_j$'s view. At the end of the game, $\mathcal{A}_1$ sends a guess bit to $\mathcal{A}_2$, which $\mathcal{A}_2$ outputs as its guess in the OT game.

We now analyze the advantage of $(\mathcal{A}_1, \mathcal{A}_2)$ pair in the OT game. If $\beta = 0$, the messages received by $\mathcal{A}_1$ is identical to hybrid $H_{t.\ell.m}$. Otherwise, it is identical to hybrid $H_{t'.\ell'.m'}$. Therefore, the pair $(\mathcal{A}_1, \mathcal{A}_2)$ has advantage $\epsilon(n)$ in distinguishing between the two cases.

■

Corollary 1. *For any $c < 1/80$, there exists an α s.t. the above k-party MPC protocol is $(O(kn), cn^2 - O(n), O(2^{-\alpha n}))$ semi-honest secure for the class of n-gate functionalities $\{\mathcal{C}_n\}_{n\in\mathbb{N}}$.*

References

1. Aumann, Y., Ding, Y.Z., Rabin, M.O.: Everlasting security in the bounded storage model. IEEE Trans. Information Theory (2002)
2. Aumann, Y., Rabin, M.O.: Information theoretically secure communication in the limited storage space model. In: CRYPTO (1999)
3. Cachin, C., Maurer, U.M.: Unconditional security against memory-bounded adversaries. In: CRYPTO (1997)
4. Ding, Y.Z., Rabin, M.O.: Hyper-encryption and everlasting security. In: STACS (2002)
5. Dodis, Y., Quach, W., Wichs, D.: Speak much, remember little: Cryptography in the bounded storage model, revisited. Cryptology ePrint Archive, Report 2021/1270 (2021), https://ia.cr/2021/1270
6. Dowsley, R., Lacerda, F., Nascimento, A.C.A.: Oblivious transfer in the bounded storage model with errors. In: IEEE International Symposium on Information Theory (2014)
7. Dziembowski, S., Maurer, U.: Tight security proofs for the bounded-storage model. In: STOC. pp. 341–350 (2002)
8. Dziembowski, S., Maurer, U.M.: On generating the initial key in the bounded-storage model. In: Cachin, C., Camenisch, J. (eds.) EUROCRYPT (2004)
9. Goldreich, O., Levin, L.A.: A hard-core predicate for all one-way functions. In: Proceedings of the Twenty-First Annual ACM Symposium on Theory of Computing. p. 2532. STOC 89, Association for Computing Machinery, New York, NY, USA (1989), https://doi.org/10.1145/73007.73010
10. Goldreich, O., Micali, S., Wigderson, A.: How to play any mental game or a completeness theorem for protocols with honest majority. In: STOC '87 (1987)
11. Grover, L.K.: A fast quantum mechanical algorithm for database search. In: STOC (1996)

12. Guan, J., Zhandary, M.: Simple schemes in the bounded storage model. In: EUROCRYPT (2019)
13. Harnik, D., Naor, M.: On everlasting security in the *Hybrid* bounded storage model. In: ICALP (2006)
14. Ishai, Y., Prabhakaran, M., Sahai, A.: Founding cryptography on oblivious transfer - efficiently. In: CRYPTO 2008 (2008)
15. Kilian, J.: Founding cryptography on oblivious transfer. In: STOC (1988)
16. Kol, G., Raz, R., Tal, A.: Time-space hardness of learning sparse parities. In: STOC (2017)
17. Lu, C.: Hyper-encryption against space-bounded adversaries from on-line strong extractors. In: CRYPTO (2002)
18. Maurer, U.M.: Conditionally-perfect secrecy and a provably-secure randomized cipher. J. Cryptology (1992)
19. Moran, T., Shaltiel, R., Ta-Shma, A.: Non-interactive timestamping in the bounded-storage model. J. Cryptology (2009)
20. Raz, R.: Fast learning requires good memory: A time-space lower bound for parity learning. In: FOCS (2016)
21. Raz, R.: A time-space lower bound for a large class of learning problems. In: FOCS (2017)
22. Savas, E., Sunar, B.: A practical and secure communication protocol in the bounded storage model. In: International Conference on Networking (2005)
23. Shikata, J., Yamanaka, D.: Bit commitment in the bounded storage model: Tight bound and simple optimal construction. In: Cryptography and Coding - IMA International Conference, IMACC (2011)
24. Shor, P.W.: Algorithms for quantum computation: Discrete logarithms and factoring. In: FOCS (1994)
25. Yao, A.: How to generate and exchange secrets. In: FOCS (1986)

Supplementary Material

A Security Proof for 1-out-of-4 OT in Bounded-Storage Model

In this section we present proof for correcness and security for the OT protocol in Section 4.

Correctness. We now prove the correctness property of the scheme in section 4.

Claim 1 *If the underlying 1-out-of-2 OT scheme is $O(n)$-correct, then the 1-out-of-4 OT scheme described above is $O(n)$-correct.*

Proof. By the correctness property of the 1-out-of-2 scheme, after step 2, the receiver will correctly compute $r^t_{t,b}$ for all $b > t$ and $r^t_{a,t}$ for all $a < t$, which are the 3 masking bits it needs to decrypt the message y_t. The receiver can then correctly decrypt y_t to get m_t. The sender and reciever runs six 1-out-of-2 OT protocols. If the underlying 1-out-of-2 OT scheme run within $O(n)$ space characteristic, then both the receiver and sender can run within $O(n)$ space.

A.1 Proof of Security

Now we prove that the construction is semi-honest secure in the bounded storage model as per Definitions 7 and 8, assuming the underlying 1-out-of-2 OT protocol is semi-honest secure in the bounded storage model. For simplicity, we denote the underlying 1-out-of-2 OT scheme by OT_2 and the 1-out-of-4 OT scheme constructed above by OT_4.

At a high level, the OT_4 scheme is secure against a semi-honest sender with bounded space, because all the message that the sender receives are part of one of the underlying OT_2 invocations. As any bounded space semi-honest sender cannot extract more information from the 1-out-of-2 protocol transcript, he cannot extract more information from the 1-out-of-4 OT protocol transcript as well.

In order to argue that the OT_4 scheme is secure against a semi-honest receiver with bounded space, let us consider an example when the receiver's choice t is 1. By the security of the OT_2 scheme, the receiver does not have any information about the bits $r^0_{(0,1)}, r^2_{(1,2)}, r^3_{(1,3)}$. As these bits are used to mask the messages m_0, m_2, m_3. The receiver cannot distinguish y_0, y_2, y_3 from uniformly random bits.

Security against Semi-Honest Sender

Theorem 8. *For any constants c, α, if OT_2 scheme is $O(n)$-correct and $(O(n), cn^2, O(2^{-\alpha n}))$-secure against semi-honest sender, then there exists corresponding constants c', α' s.t. the above OT_4 scheme is $(O(n), c'n^2, O(2^{-\alpha' n}))$-secure against semi-honest sender.*

Proof. We first construct a simulator $\mathsf{OT}_4.\mathsf{SenderSim}_4$ which outputs a simulated view of the sender in OT_4 protocol, given input $\{m_j\}_{j\in[3]}$. The simulator outputs the view via its write communication tape.

$\mathsf{SenderSim}_4(1^n, (m_0, m_1, m_2, m_3))$:
Write $(\mathsf{Input}, (m_0, m_1, m_2, m_3))$ on the write communication tape.
For each pair $(a, b) \in \{0, 1, 2, 3\}^2$ s.t. $a < b$:
- Sample uniformly random $(r^a_{a,b}, r^b_{a,b}) \leftarrow \{0, 1\}^2$, and write $(\mathsf{sample}, r^a_{a,b})$, $(\mathsf{sample}, r^b_{a,b})$ to the write tape.
- Run $\mathsf{OT}_2.\mathsf{SenderSim}_2(1^n, (r^a_{a,b}, r^b_{a,b}))$, and write the generated view on the write tape.

The OT_4 simulator internally invokes OT_2 simulator 6 times sequentially. As the OT_2 simulator has space characteristic $O(n)$, the OT_4 simulator also has space characteristic $O(n)$. We now prove that a space bounded adversary cannot distinguish between the views generated by the real protocol and the simulator via a hybrid argument. First let us order the six tuples $(a, b) \in \{0, 1, 2, 3\}^2$ s.t. $a < b$ in any order from 1 to 6. Let the order of any tuple (a, b) be $\mathsf{Order}(a, b)$.

At a high level, in the hybrid H_i, for the tuples with order less than or equal to i, a real OT_2 protocol is run internally to generate the corresponding view, whereas for the tuples with order more than i, $\mathsf{OT}_2.\mathsf{SenderSim}_2$ is used to generate the view of the corresponding 1-out-of-2 OT. The description of the algorithm used in H_i is formally described below. Note that unlike the $\mathsf{SenderSim}_4$, these intermediate hybrids additionally take the receiver's choice t as input.

$H_i(1^n, (m_0, m_1, m_2, m_3), t)$ $(0 \le i \le 6)$:
Write $(\mathsf{Input}, (m_0, m_1, m_2, m_3))$ on the write communication tape.
For each $(a, b) \in \{0, 1, 2, 3\}^2$ s.t. $a < b$,
- Sample uniformly random $(r^a_{a,b}, r^b_{a,b}) \leftarrow \{0, 1\}^2$, and write $(\mathsf{sample}, r^a_{a,b})$, $(\mathsf{sample}, r^b_{a,b})$ to the write tape.
- If $\mathsf{Order}(a, b) \le i$, If $t = a$, set $c = 0$. If $t = b$, set $c = 1$. Otherwise, sample $c \leftarrow \{0, 1\}$. Internally run the OT_2 protocol with sender's input $(r^a_{a,b}, r^b_{a,b})$, and the receiver's input c. Throughout the protocol, write the sender's transcript on the write tape.
- If $\mathsf{Order}(a, b) > i$, run OT_2 simulator $\mathsf{SenderSim}_2(1^n, (r^a_{a,b}, r^b_{a,b}))$, and write the generated view on the write tape.

Clearly, the distribution of the view generated by Hybrid H_0 is identical to the view generated in the real protocol. Similarly, the distribution of the view generated by final hybrid H_6 is identical to the view generated by the simulator $\mathsf{OT}_4.\mathsf{SenderSim}_4$. We now show that if a space-bounded adversary can distinguish between the hybrids H_i and H_{i+1} with an ϵ advantage, then there exists a space-bounded reduction algorithm that can break the 1-out-of-2 OT security with ϵ advantage. Formally we state the lemma below. Consider any adversary $\mathcal{A}$ with 1 read-tape. For any hybrid H_i, we let $\Pr[H_i^{\mathcal{A}}(1^n, (m_0, m_1, m_2, m_3), t) = 1]$ to be the probability that the adversary outputs 1 when the write tape of $H_i(1^n, (m_0, m_1, m_2, m_3), t)$ algorithm is connected to the read tape of the adversary.

Lemma 3. *For any constants c, α, if the underlying OT_2 scheme is $O(n)$-correct and $(O(n), cn^2, O(2^{-\alpha n}))$-secure against semi-honest sender, then there exists corresponding constants c', α' s.t. for every $c'n^2$-space bounded adversary $\mathcal{A}$, there exists an integer N_0 s.t. for any space parameter $n > N_0$, any sender's input tuple (m_0, m_1, m_2, m_3), receiver's choice t, any index $0 \leq i < 6$, $|\Pr[H_i^{\mathcal{A}}(1^n, (m_0, m_1, m_2, m_3), t) = 1] - \Pr[H_{i+1}^{\mathcal{A}}(1^n, (m_0, m_1, m_2, m_3), t) = 1]| \leq O(2^{-\alpha' n})$.*

Proof. We show that if for all c', α' s.t. there is a $c'n^2$-space bounded adversary $\mathcal{A}$, for all integer N_0 s.t. there exists space parameter $n > N_0$, some sender's input tuple (m_0, m_1, m_2, m_3), some receiver's choice t, some index $0 \leq i < 6$, $|\Pr[H_i^{\mathcal{A}}(1^n, (m_0, m_1, m_2, m_3), t) = 1] - \Pr[H_{i+1}^{\mathcal{A}}(1^n, (m_0, m_1, m_2, m_3), t) = 1]| \geq \Omega(2^{-\alpha' n})$; then there exists constants c, α, such that there is a cn^2-space characteristic semi-honest sender adversary $\mathcal{A}'$ that breaks the underlying OT_2 scheme with $\Omega(2^{-\alpha n})$ advantage.

Let the challenger in hybrid H_i be an adversary $\mathcal{A}'$ in an OT_2 security game against the semi-honest sender. In the reduction during H_i, for all ordered tuples (a, b) where $\mathsf{Order}(a, b) < i$, $\mathcal{A}'$ runs a real OT_2 protocol as described above and writes the sender's view on the read tape of adversary $\mathcal{A}$; for $k = i + 1$, $\mathcal{A}'$ receives the sender adversary's view from the challenger in $\mathsf{GameSHSender}^{\mathcal{A}}_{(r^a_{a,b}, r^b_{a,b}), c}$ against semi-honest sender, where $\mathsf{Order}(a, b) = i + 1$; $\mathcal{A}'$ writes this view on the read tape of $\mathcal{A}$; for $k > i + 1$, $\mathcal{A}'$ runs $\mathsf{SenderSim}_2$ as described above for H_i.

At the end of the game, $\mathcal{A}$ outputs a bit $b' \in \{0, 1\}$ and $\mathcal{A}'$ passes it as its own output to the challenger in $\mathsf{GameSHSender}^{\mathcal{A}}_{(r^a_{a,b}, r^b_{a,b}), c}$, where $\mathsf{Order}(a, b) = i+1$. $\mathcal{A}'$ has advantage $|\Pr[H_i^{\mathcal{A}}(1^n, (m_0, m_1, m_2, m_3), t) = 1] - \Pr[H_{i+1}^{\mathcal{A}}(1^n, (m_0, m_1, m_2, m_3), t) = 1]|$. Hence if there exists $\mathcal{A}$ that has $\Omega(2^{-\alpha' n})$ advantage difference between two hybrids, $\mathcal{A}'$ has $\Omega(2^{-\alpha' n})$ advantage in winning $\mathsf{GameSHSender}^{\mathcal{A}}_{(r^a_{a,b}, r^b_{a,b}), c}$ against semi-honest sender.

By triangle inequality, the advantage of adversary $\mathcal{A}$ in the OT_4 game $\mathsf{GameSH}^{\mathcal{A}}_{\{m_j\}_{j \in [3]}, t}$ is

$$|\Pr[H_0^{\mathcal{A}}(1^n, (m_0, m_1, m_2, m_3), t) = 1] - \Pr[H_6^{\mathcal{A}}(1^n, (m_0, m_1, m_2, m_3), t) = 1]|$$

$$\leq \sum_{i=0}^{5} |\Pr[H_i^{\mathcal{A}}(1^n, (m_0, m_1, m_2, m_3), t) = 1] - \Pr[H_{i+1}^{\mathcal{A}}(1^n, (m_0, m_1, m_2, m_3), t) = 1]|$$

If there exists $\mathcal{A}$ that has $\Omega(2^{-\alpha n})$ advantage in $\mathsf{GameSHSender}^{\mathcal{A}}_{\{m_j\}_{j \in [3]}, t}$, then there must exist some α' such that some adversary $\mathcal{A}'$ has advantage $\Omega(2^{-\alpha' n}) = \Omega(2^{-\alpha n}/6)$ in OT_2 semi-honest sender security game on some input $((r^a_{a,b}, r^b_{a,b}), c)$, which contradicts the proved security of OT_2.

Corollary 2. *For any $c < 1/40$, there exists $\alpha > 0$ such that the above 1-out-of-4 OT construction is $(O(n), c \cdot n^2, O(2^{-\alpha n}))$-secure against semi-honest sender.*

The corollary follows when the above OT_4 scheme is used with the OT_2 construction described in Appendix B.

Security against Semi-Honest Receiver

Theorem 9. *If the underlying* OT_2 *scheme is* $O(n)$*-correct and* $(O(n), \infty, O(2^{-n}))$*-secure against semi-honest receiver, then the above 1-out-of-4 OT construction is* $(O(n), \infty, O(2^{-n}))$*-secure against semi-honest receiver.*

Proof. We first construct a receiver simulator for OT_4, $\mathsf{ReceiverSim}_4$ which outputs a simulated view of the receiver in the OT_4 protocol, given the receiver's choice $t \in \{0,1,2,3\}$ and the receiver's output bit m_t. The simulator outputs the view via its write communication tape.

$\mathsf{ReceiverSim}_4(1^n, (t, m_t))$:

- Write $(\mathsf{Input}, (t, m_t))$ to the write communication tape.
- For each pair $(a, b) \in \{0,1,2,3\}^2$ s.t. $a < b$:
  - If $t = a$, sample $r^a_{a,b} \leftarrow \{0,1\}$, run $\mathsf{OT}_2.\mathsf{ReceiverSim}_2(1^n, (0, r^a_{a,b}))$.
  - If $t = b$, sample $r^b_{a,b} \leftarrow \{0,1\}$, run $\mathsf{OT}_2.\mathsf{ReceiverSim}_2(1^n, (1, r^b_{a,b}))$.
  - Else, sample $c \leftarrow \{0,1\}$ and write (sample, c) to the write tape.
    - If $c = 0$, sample $r^a_{a,b} \leftarrow \{0,1\}$ and run $\mathsf{OT}_2.\mathsf{ReceiverSim}_2(1^n, (0, r^a_{a,b}))$.
    - If $c = 1$, sample $r^b_{a,b} \leftarrow \{0,1\}$ and run $\mathsf{OT}_2.\mathsf{ReceiverSim}_2(1^n, (1, r^b_{a,b}))$.
  - In all the cases, copy the view generated by OT_2 simulator to the write tape.
- Sample $y_j \leftarrow \{0,1\}$ for $j \neq t$. Compute y_t by encrypting m_t as in the real game. Write $(\mathsf{receive}, (y_0, y_1, y_2, y_3))$ on the write tape.

The OT_4 simulator internally invokes OT_2 simulator 6 times sequentially. As the OT_2 simulator has space characteristic $O(n)$, the OT_4 simulator also has space characteristic $O(n)$, the OT_4 simulator also has space characteristic $O(n)$. We now prove that a space bounded adversary cannot distinguish between the views generated by the real protocol and the simulator via a hybrid argument. Recall that we order all possible tuples in $\{0,1,2,3\}^2$ by numerical order and denote that the k-th tuple has $\mathsf{Order}(a, b) = k$.

At a high level, in the hybrid H_i, for the tuples with order less than or equal to i, a real OT_2 protocol is run internally to generate the corresponding view; whereas for the tuples with order more than i, $\mathsf{OT}_2.\mathsf{ReceiverSim}_2$ is used to generate the view of the corresponding 1-out-of-2 OT.

The distribution of the view generated by Hybrid H_0 is identical to the view generated in the real protocol. Similarly, the distribution of the view generated by the final hybrid H_6 is identical to the view generated by the simulator $\mathsf{OT}_4.\mathsf{ReceiverSim}_4$. We now show that if an adversary can distinguish between the hybrids H_i and H_{i+1} with an ϵ advantage, then there exists a space-bounded reduction algorithm that can break the 1-out-of-2 OT security against semi-honest receiver with ϵ advantage. Formally we state the lemma below. Consider any adversary $\mathcal{A}$ with 1 read-tape. For any hybrid H_i, we let $\Pr[H_i^{\mathcal{A}}(1^n, (m_0, m_1, m_2, m_3), t) = 1]$ to be the probability that the adversary outputs 1 when the write tape of $H_i(1^n, (m_0, m_1, m_2, m_3), t)$ algorithm is connected to the read tape of the adversary.

Lemma 4. *If the underlying* OT_2 *scheme is* $O(n)$*-correct and* $(O(n), \infty, O(2^{-n}))$*-secure against semi-honest receiver, then for every adversary* $\mathcal{A}$*, there exists an integer* N_0 *s.t. for any space parameter* $n > N_0$*, any sender's input tuple* (m_0, m_1, m_2, m_3)*, receiver's choice* t*, for any index* $0 \leq i < 6$*,* $\Pr[H_i^{\mathcal{A}}(1^n, (m_0, m_1, m_2, m_3), t) = 1] - \Pr[H_{i+1}^{\mathcal{A}}(1^n, (m_0, m_1, m_2, m_3), t) = 1]| \leq O(2^{-n})$.

Proof. We show that if there is some adversary $\mathcal{A}$, for all integer N_0 s.t. there exists space parameter $n > N_0$, some sender's input tuple (m_0, m_1, m_2, m_3), some receiver's choice t, some index $0 \leq i < 6$, $|\Pr[H_i^{\mathcal{A}}(1^n, (m_0, m_1, m_2, m_3), t) = 1] - \Pr[H_{i+1}^{\mathcal{A}}(1^n, (m_0, m_1, m_2, m_3), t) = 1]| \geq \Omega(2^{-\alpha' n})$; then there exists some semi-honest receiver adversary $\mathcal{A}'$ that breaks the underlying OT_2 scheme with $\Omega(2^{-n})$ advantage.

Let the challenger in hybrid H_i be an adversary $\mathcal{A}'$ in an OT_2 security game against the semi-honest receiver. In the reduction during H_i, for all ordered tuples (a, b) where $\mathsf{Order}(a, b) \leq i$, $\mathcal{A}'$ runs a real OT_2 protocol as described above; for $\mathsf{Order}(a, b) = i + 1$, $\mathcal{A}'$ gets the adversary's view from the challenger in $\mathsf{GameSHReceiver}^{\mathcal{A}}_{(r^a_{a,b}, r^b_{a,b}), c}$ against semi-honest receiver, and then $\mathcal{A}'$ writes this view on the read tape of $\mathcal{A}$; for $\mathsf{Order}(a, b) > i + 1$, $\mathcal{A}'$ runs $\mathsf{ReceiverSim}_2$ as described above for H_i.

At the end of the game, $\mathcal{A}$ outputs a bit $b' \in \{0, 1\}$ and $\mathcal{A}'$ passes it as its own output to the challenger in $\mathsf{GameSHReceiver}^{\mathcal{A}}_{(r^a_{a,b}, r^b_{a,b}), c}$, $\mathsf{Order}(a, b) = i+1$. $\mathcal{A}'$ has advantage $|\Pr[H_i^{\mathcal{A}}(1^n, (m_0, m_1, m_2, m_3), t) = 1] - \Pr[H_{i+1}^{\mathcal{A}}(1^n, (m_0, m_1, m_2, m_3), t) = 1]|$. Hence if there exists $\mathcal{A}$ that can distinguish between two hybrids with probability $\Omega(2^{-n})$, $\mathcal{A}'$ has $\Omega(2^{-n})$ advantage in winning $\mathsf{GameSHReceiver}^{\mathcal{A}}_{(r^a_{a,b}, r^b_{a,b}), c}$ against semi-honest receiver.

Conclusion. By triangle inequality, the advantage of adversary $\mathcal{A}$ in the OT_4 game $\mathsf{GameSH}^{\mathcal{A}}_{\{m_j\}_{j\in[3]}, t}$ is given by

$$|\Pr[H_0^{\mathcal{A}}(1^n, (m_0, m_1, m_2, m_3), t) = 1] - \Pr[H_6^{\mathcal{A}}(1^n, (m_0, m_1, m_2, m_3), t) = 1]|$$

$$\leq \sum_{i=0}^{5} |\Pr[H_i^{\mathcal{A}}(1^n, (m_0, m_1, m_2, m_3), t) = 1] - \Pr[H_{i+1}^{\mathcal{A}}(1^n, (m_0, m_1, m_2, m_3), t) = 1]|$$

If there exists $\mathcal{A}$ that has $\Omega(2^{-n})$ advantage in $\mathsf{GameSHReceiver}^{\mathcal{A}}_{\{m_j\}_{j\in[3]}, t}$, then there must exist some adversary $\mathcal{A}'$ has advantage $\Omega(2^{-n}) = \Omega(2^{-n}/6)$ in OT_2 semi-honest receiver security game on some input $((r^a_{a,b}, r^b_{a,b}), c)$, which contradicts the proved security of OT_2.

B Review of 1-out-of-2 Semi-honest OT Protocol

In this section, we recall the 1-out-of-2 OT construction by Guan and Zhandry [12] and prove its security. [12] proposed an indistinguishability based security definition for 1-out-of-2 OT, whereas we need simulation-based secure OT in our

construction of MPC protocol. We thereby prove the security of their construction as per Definitions 7 and 8.

At a high level, the OT construction proceeds as follows. Let (x_0, x_1) be the sender's input message bits, and let d be the receiver's choice bit. The receiver sends a randomly sampled stream of parity learning tuples $(a_1, b_1), \cdots (a_\ell, b_\ell)$ to the sender. The sender stores two random linear combinations (L_0, q_0) and (L_1, q_1) of these samples. For a sufficiently large m, these samples stored by the sender statistically resemble a fresh parity learning samples. The receiver then encrypts its choice bit d i.e., creates a fresh parity learning sample (a, b) and sends $(a, b + d)$ to the sender. The sender then computes encryptions of $(1 - d) \cdot x_0$ and $d \cdot x_1$, rerandomizes these by adding (L_0, q_0) and (L_1, q_1) and sends them to the receiver. The receiver can decrypt $(1 - d) \cdot x_0$ and $d \cdot x_1$ using its secret key and obtain x_d.

We now describe the protocol formally. The construction is parameterized by a space parameter n. Intuitively, the honest parties can run the protocol within space characteristic $O(n)$, whereas the dishonest parties with space characteristic $O(n^2)$ cannot break the security of the protocol.

1. The receiver samples a random key $\mathbf{k} \leftarrow \{0,1\}^n$. The sender sets $\mathbf{L} = \{0\}^{2\times n}$ and $\mathbf{q} \leftarrow [0, 0]^\top$. Let $\ell = 2n$.
2. For $i = 1$ to ℓ,
   The receiver samples $\mathbf{a}_i \leftarrow \{0,1\}^n$ and sends $(\mathbf{a}_i, \mathbf{a}_i \cdot \mathbf{k} \bmod 2)$ to the sender.
   Upon receiving $(\mathbf{a}_i, b_i)$, the sender first samples $\mathbf{M}_i \leftarrow \{0,1\}^2$ and updates $\mathbf{L} \leftarrow \mathbf{L} + \mathbf{M}_i \cdot \mathbf{a}_i$ and $\mathbf{q} \leftarrow \mathbf{q} + \mathbf{M}_i \cdot b_i$, where $\mathbf{a}_i$ is interpreted as a row vector and $\mathbf{M}_i$ is interpreted as a column vector.
3. The receiver then samples a random $\mathbf{y} \leftarrow \{0,1\}^n$ and sends a tuple $(\mathbf{y}, \mathbf{y} \cdot \mathbf{k} + d)$ to the sender.
4. Let us denote $\mathbf{L}_0, \mathbf{L}_1$ as the first and second rows of $\mathbf{L}$, and denote q_0, q_1 to be the first and second element in $\mathbf{q}$. The sender sends the following encryptions of $(1 - d)x_0, dx_1$ to the receiver:

   $$(\mathbf{L}_0, q_0) + x_0 \cdot (\mathbf{y}, \mathbf{y} \cdot \mathbf{k} + 1 - d) = (\mathbf{L} + x_0 \cdot \mathbf{y}, (\mathbf{L} + x_0 \cdot \mathbf{y}) \cdot \mathbf{k} + x_0(1 - d))$$

   and

   $$(\mathbf{L}_1, q_1) + x_1 \cdot (\mathbf{y}, \mathbf{y} \cdot \mathbf{k} + d) = (\mathbf{L} + x_1 \cdot \mathbf{y}, (\mathbf{L} + x_1 \cdot \mathbf{y}) \cdot \mathbf{k} + x_1 d).$$

5. Let the values received by the receiver be $(c_0, c_0') \in \{0,1\}^n \times \{0,1\}$ and $(c_1, c_1') \in \{0,1\}^n \times \{0,1\}$. The receiver computes $c_0' - c_0 \cdot k \bmod 2$ and $c_1' - c_1 \cdot k \bmod 2$. If both values are 0, the receiver outputs 0. Otherwise, the receiver outputs 1.

Correctness. Note that in the last step, $c_0' - c_0 \cdot k \bmod 2$ evaluates to $x_0 \cdot (1 - d)$, and $c_1' - c_1 \cdot k \bmod 2$ evaluates to $x_1 \cdot d$. In case the chosen message $x_d = 0$, then both the values are 0 and the receiver outputs 0. Whereas if $x_d = 1$, then one of these values is 1, and the receiver outputs 1.

The above protocol can be run within $O(n)$ space characteristic. This is because, at each step of the protocol, the receiver has to store the key k and at most two other n-bit messages. The sender only maintains $\mathbf{L}, q$ which also requires only $O(n)$ space.

B.1 Proof of Security

Theorem 10. *For any $c < 1/40$, there exists $\alpha > 0$ such that the above 1-out-of-2 OT construction is $(O(n), c \cdot n^2, O(2^{-\alpha n}))$-secure against semi-honest sender.*

Proof. We first build a simulator $\mathsf{SenderSim}$ which outputs the simulated view of the sender given its input messages (x_0, x_1). The simulator outputs the view via its write communication tape.

$\mathsf{SenderSim}(1^n, (x_0, x_1))$:
- Place $(\mathsf{Input}, (x_0, x_1))$ on the write comm. tape.
- Sample $\mathbf{k} \leftarrow \{0,1\}^n$.
- For $i = 1$ to $2n$,
  Sample $\mathbf{a}_i \leftarrow \{0,1\}^n$, compute $b_i = \mathbf{a}_i \cdot \mathbf{k} \bmod 2$, and write $(\mathsf{receive}, (\mathbf{a}_i, b_i))$ on the write tape.
  Sample $\mathbf{M}_i \leftarrow \{0,1\}^2$, and write $(\mathsf{sample}, \mathbf{M}_i)$ on the write tape.
- Sample $\mathbf{y} \leftarrow \{0,1\}^n$ and a bit $r \leftarrow \{0,1\}$, and write $(\mathsf{receive}, (\mathbf{y}, r))$ on the write tape.

To prove Theorem 10, we prove that if there exists a space bounded semi-honest sender that can break the security defined above, then we can construct a space bounded adversary that breaks the indistinguishability version of parity learning, defined in 9 in this paper.

For the sake of contradiction, suppose there exists some $c < 1/40$, for all $\alpha > 0$, there exists a cn^2-space bounded adversary such that for all $N_0 \in \mathbb{N}$, there is some $n > N_0, \{x_i\}_{i\in[k]} \subseteq \mathcal{M}, d \in \{0,1\}$ where $\mathcal{A}$ has advantage $\Omega(2^{-\alpha n})$ in the $\mathsf{GameSHSender}^{\mathcal{A}}_{\{x_i\}_{i\in[1]},c}$ security game; then for all $\alpha' > 0$, there is a $(2cn^2 + O(n))$-space bounded adversary $\mathcal{A}'$ where $\mathcal{A}'$ has advantage $\Omega(2^{-\alpha' n})$ in the indistinguishability parity learning security game. There exists some N_0, such that for all $n > N_0$, there exists some $c' < 1/20$ such that $c'n^2 \geq 2cn^2 + O(n)$.

The challenger in the semi-honest sender security game $\mathsf{GameSHSender}^{\mathcal{A}}_{\{x_j\}_{j\in[1]},c}$ is an adversary $\mathcal{A}'$ in the indistinguishability parity learning security game denoted as $\mathsf{PL}_{\mathcal{A}',\delta}(n, 2n+1)$, where the secret key of parity learning has length n and the adversary will receive $2n$ parity learning tuples plus one challenge tuple.

If $\mathcal{A}'$'s coin flip is $b = 1$, $\mathcal{A}'$ runs the real-world OT_2 protocol, regardless of the $\mathsf{PL}_{\mathcal{A}',\delta}(n, 2n+1)$ challenger. If $\mathcal{A}'$'s coin it flips $b = 0$, then $\mathcal{A}'$ writes the parity learning tuples from $\mathsf{PL}_{\mathcal{A}',\delta}(n, 2n+1)$ on communication tape of the simulator. In the reduction, the simulator in $\mathsf{GameSHSender}^{\mathcal{A}}_{\{x_j\}_{j\in[1]},d}$ does not sample $\mathbf{k}, \mathbf{r}_i, \mathbf{y}$ by itself but instead gets $(\mathbf{r}_i, a_i), i \in [2n+1]$ from the $\mathsf{PL}_{\mathcal{A},\delta}(n, 2n+1)$ game. For the first $2n$ tuples $(\mathbf{a}_i, b_i), i \in [2n]$, the simulator uses them as the tuples in

protocol step 2; when $i = 2n + 1$, the sender simulator sets $\mathbf{y} = \mathbf{a}_{2n+1}$ and sets $r = b_{2n+1}$.

Recall that when $i = 2n + 1$, the $\mathsf{PL}_{\mathcal{A}',b}(n, 2n + 1)$ challenger flips a coin $\delta \leftarrow \{0, 1\}$; it sends $b_{2n+1} = \mathbf{a}_{2n+1} \cdot \mathbf{k}$ to $\mathcal{A}'$ if $\delta = 0$ and sends uniformly random $b_{2n+1} \leftarrow \{0, 1\}$ to the $\mathcal{A}'$ if $\delta = 1$. At the end of the game, $\mathcal{A}$ outputs a bit b'; $\mathcal{A}'$ passes $\mathcal{A}$'s output b' to the the $\mathsf{PL}_{\mathcal{A}',b}(n, 2n+1)$ challenger as its own output δ'. $\mathcal{A}'$ has space characteristic $(cn^2+O(n))$ where cn^2 space is used to simulate the $\mathcal{A}$ and $O(n)$ for interaction with challenger. Suppose $\mathcal{A}$ has advantage $\epsilon = \Omega(2^{-\alpha n})$, then $\mathcal{A}'$ has advantage $\epsilon/2 = \Omega(2^{-\alpha' n})$ in $\mathsf{PL}_{\mathcal{A}',b}(n, 2n + 1)$, which forms a contradiction with 6.

Theorem 11. *The above 1-out-of-2 OT construction is $(O(n), \infty, O(2^{-n}))$-secure against semi-honest receiver.*

Proof. We first build a simulator $\mathsf{ReceiverSim}$ which outputs the simulated view of the receiver given the choice bit d and the receiver's output x_d. The simulator outputs the view via its write communication tape.

$\mathsf{ReceiverSim}(1^n, (d, x_d))$:
- Write $(\mathsf{Input}, (d, x_d))$ to the write comm. tape.
- Sample key $\mathbf{k} \leftarrow \{0, 1\}^n$, and place $(\mathsf{sample}, \mathbf{k})$ on the write tape.
- For $i = 1$ to $2n$, sample key $\mathbf{a}_i \leftarrow \{0, 1\}^n$, and write $(\mathsf{sample}, \mathbf{a}_i)$ on the write tape.
- Sample $\mathbf{y} \leftarrow \{0, 1\}^n$, and place $(\mathsf{sample}, \mathbf{k})$ on the write tape.
- Sample uniformly random $\mathbf{U} \leftarrow \{0, 1\}^{2*n}$. Let $\mathbf{U}_0, \mathbf{U}_1$ be the first and second row vectors.
- If $d = 0$:
  Write $(\mathsf{receive}, (\mathbf{U}_0, \mathbf{U}_0 \cdot \mathbf{k} + x_0))$ and $(\mathsf{receive}, (\mathbf{U}_1, \mathbf{U}_1 \cdot \mathbf{k}))$ on the write tape.
  else if $d = 1$:
  Write $(\mathsf{receive}, (\mathbf{U}_0, \mathbf{U}_0 \cdot \mathbf{k}))$ and $(\mathsf{receive}, (\mathbf{U}_1, \mathbf{U}_1 \cdot \mathbf{k} + x_1))$ on the write tape.

Notice that other than the final step, rest of the view generated by the simulator is identical to the view of the receiver in the real protocol. Suppose $d = 0$. In the final step, the receiver in the real protocol receives $(\mathbf{L}_0 + x_0 \cdot \mathbf{y}, (\mathbf{L}_0 + x_0 \cdot \mathbf{y}) \cdot \mathbf{k} + x_0)$, $(\mathbf{L}_1 + x_1 \cdot \mathbf{y}, (\mathbf{L}_1 + x_1 \cdot \mathbf{y}) \cdot \mathbf{k} + 0)$, where $\mathbf{L}_0, \mathbf{L}_1$ are random linear combinations of $\mathbf{a}_i$ vectors. Note that the stream of randomness $\{\mathbf{a}_i\}_{i\in[2n]}$ can be viewed as a $2n \times n$ matrix, and with all but exponentially small probability, the matrix has rank n. Therefore, a random linear combination of these rows $(\mathbf{L}_0, \mathbf{L}_1)$ are statistically indistinguishable from uniform sampled vectors in $\{0, 1\}^n$. Consequently, $(\mathbf{L}_0 + x_0 \cdot \mathbf{y}, (\mathbf{L}_0 + x_0 \cdot \mathbf{y}) \cdot \mathbf{k} + x_0)$, $(\mathbf{L}_1 + x_1 \cdot \mathbf{y}, (\mathbf{L}_1 + x_1 \cdot \mathbf{y}) \cdot \mathbf{k} + 0)$ are statistically indistinguishable from $(\mathbf{U}_0, \mathbf{U}_0 \cdot \mathbf{k} + x_0)$ and $(\mathbf{U}_1, \mathbf{U}_1 \cdot \mathbf{k})$. Similar analysis works even when $d = 1$.