On the Message Complexity of Secure Multiparty Computation

Yuval Ishai[1], Manika Mittal[2], and Rafail Ostrovsky[3]

[1] Technion, yuvali@cs.technion.ac.il
[2] UCLA and Yahoo!, manikamittal22@gmail.com
[3] UCLA, rafail@cs.ucla.edu

Abstract. We study the minimal number of point-to-point messages required for general secure multiparty computation (MPC) in the setting of computational security against semi-honest, static adversaries who may corrupt an arbitrary number of parties.
We show that for functionalities that take inputs from n parties and deliver outputs to k parties, $2n+k-3$ messages are necessary and sufficient. The negative result holds even when given access to an arbitrary correlated randomness setup. The positive result can be based on any 2-round MPC protocol (which can in turn can be based on 2-message oblivious transfer), or on a one-way function given a correlated randomness setup.

1 Introduction

Since the seminal works from the 1980s that established the feasibility of secure multiparty computation (MPC) [24, 19, 3, 9], there has been a large body of work on different *efficiency* measures of MPC protocols. In particular, a lot of research efforts were aimed at characterizing the minimal communication complexity, round complexity, computational complexity, and randomness complexity of MPC protocols.

In the present work we study the *message complexity* of MPC protocols, namely the number of messages that the parties need to communicate to each other over point-to-point channels. While there have been a few prior works studying the message complexity of MPC in different settings (see Section 1.2 below), this complexity measure received relatively little attention. The goal of minimizing the message complexity of protocols is motivated by scenarios in which sending or receiving a message has a high cost, which is not very sensitive to the size of the message. For instance, this is the case when using a traditional postal system for message delivery (say, shipping optical media from one party to another), or when establishing a communication channel between pairs of parties is expensive due to limited connectivity.

The main focus of our work is on the standard model of *computationally* secure MPC in the presence of a *static* (non-adaptive), *semi-honest* (passive) adversary, who may corrupt an *arbitrary subset* of the parties. In this model, we ask the following question:

> *How many messages are needed for securely computing functions that take inputs from n parties and deliver outputs to k of these parties?*

For simplicity, we consider the above question in the setting of fixed, or "oblivious," interaction patterns, a commonly used assumption in the MPC literature (see, e.g., [11, 20]). In this setting, it is assumed that the protocol specifies a-priori the sender-receiver pairs of the messages sent in each round.[1]

1.1 Our Contribution

Our main result is a sharp answer to the above question: we show that in the setting discussed above, $2n + k - 3$ messages are necessary and sufficient.

The negative result holds even when the parties can communicate over *secure* point-to-point channels or, more generally, even when allowing an arbitrary input-independent correlated randomness setup. This result builds (non-trivially) on the general characterization of the power MPC with general interaction patterns from the recent work of Halevi et al. [20].

The positive result can be based on any 2-round MPC protocol, applying a natural greedy message forwarding strategy to emulate the quadratic number of messages of such protocols with an optimal number of messages. Using recent constructions of 2-round MPC protocols, this approach can be instantiated in the plain model, public point-to-point channels, under the (minimal) assumption that a 2-message semi-honest oblivious transfer protocol exists [4, 17]. (Alternative constructions with incomparable efficiency features can be based on the LWE assumption [23] or even the DDH assumption given a PKI setup [6]). Given a general correlated randomness setup, the positive result can be based on any one-

[1] Message complexity is more subtle when allowing dynamic interaction patterns, since not receiving a message also conveys information; see e.g. [13] for discussion. Our positive results do not require this relaxation. Moreover, our negative result can be extended to capture dynamic interactions, by exploiting the fact that the adversary can "guess" the identity of a party that sends a constant number of messages with high success probability and corrupt all of the other parties.

way function, or even provide unconditional information theoretic security when considering low-complexity functions such as NC^1 functions.

1.2 Related Work

As mentioned above, Halevi et al. [20] consider the question of MPC with general interaction patterns, giving a full characterization for the "best possible security" of an MPC protocol that uses a given interaction pattern with a general correlated randomness setup. Our negative result builds on their general characterization, focusing on the case where the "best possible security" coincides with the standard notion of security. The positive results in [20] consider a more general setting that (inevitably) requires the use of indistinguishability obfuscation and a correlated randomness setup. In contrast, our positive results rely on weaker assumptions and apply also to the plain model.

The message complexity of MPC protocols has been explicitly considered in several previous works, but the model of MPC considered in these works is quite different from ours. In particular, the message complexity in the *information-theoretic* setting with a *bounded fraction* of corrupted parties has been studied in [11, 7, 5, 13, 14]. Our focus on computational security (or alternatively, allowing a correlated randomness setup) allows us to circumvent previous lower bounds that apply to the information-theoretic setting. In particular, our positive results circumvent the quadratic message lower bound from [11]. On the other hand, considering an adversary that can corrupt an arbitrary number of parties rules out MPC protocols that achieve sublinear message complexity in the number of parties by assigning the computation to a small random subset of parties (see, e.g., [12, 7, 16]).

Organization. Following some preliminaries (Section 2), we present our negative result in Section 3 and our positive results in Section 4. In Appendix A we include a standard definition of MPC for self-containment.

2 Preliminaries

By default, we consider an MPC protocol Π for an n-party functionality f to provide *computational* security against a *semi-honest* adversary that may *statically* (non-adaptively) corrupt an arbitrary subset of the parties and eavesdrop on all communication channels. That is, the communication takes place over public point-to-point channels.

We also consider MPC with *correlated randomness setup*, where the parties are given access to a trusted source of (input-independent) correlated randomness. Note that correlated randomness setup trivially allows secure point-to-point communication over public communication channels. Thus, since our negative result applies also to this model, it applies in particular for protocols over secure point-to-point channels.

As is typically the case for security against semi-honest adversaries, our results are quite insensitive to the details of the model beyond those mentioned above. We refer to reader to Appendix A or to [18] for a standard formal treatment of MPC in this model.

3 The Lower Bound

In this section, we prove our main lower bound: in any n-party MPC protocol for computing a function with $k \geq 1$ outputs, the number of point-to-point messages is at least $2n + k - 3$. This lower bound holds even in the setting of security against semi-honest adversaries and even when the parties are given access to an arbitrary trusted source of (input-independent) correlated randomness.

The work of Halevi et al. [20] gives a general characterization for the "best possible security" of an MPC protocol with general correlated randomness setup and a given interaction pattern. The characterization in [20] is mainly intended for the case of limited interactions that warrant a relaxed notion of MPC security, and is only formulated for the case of protocols that deliver output to a single party. Here we give a simple self-contained treatment for the case of standard MPC security with an arbitrary number of outputs.

We start by defining a simplified notion of an interaction pattern, which specifies an ordered sequence of pairs of parties that represent the sender and receiver of each message. Note that we implicitly assume here that the protocol sends only a single message in each round. However, any protocol can be trivially converted into this form by splitting the messages sent in each round into multiple rounds in an arbitrary order.

Definition 3.1 (Interaction pattern). *An n-party interaction pattern is specified a sequence of pairs $M \in ([n] \times [n])^*$. The* length *of M is the number of pairs in the sequence. We say that an n-party MPC protocol Π complies with an n-party interaction pattern $M = ((a_1, b_1), \ldots, (a_m, b_m))$ if for every $1 \leq i \leq m$, the communication in Round i of Π involves only a single message, sent from party P_{a_i} to party P_{b_i}.*

It is convenient to represent an interaction pattern M by a directed (multi-)graph, whose nodes represent parties and whose edges represent messages sent over point-to-point channels. Each edge is labeled by its index in M. A *trail* in the graph is a (non-simple, directed) path that respects the order of edges and can visit the same node more than once. We formalize this below.

Definition 3.2 (Interaction graph). *Let $M = ((a_1, b_1), \ldots, (a_m, b_m))$ be an n-party interaction pattern. We let G_M denote the labeled directed multi-graph whose node set is $[n]$ and whose edges form the sequence $(e_1, \ldots, e_m)$ where $e_i = (a_i, b_i)$. (Each edge e_i in G_M is labeled by its index i.) A* trail *from node u to node v in G_M is a sequence of edges $(e_{i_1}, \ldots, e_{i_\ell})$ such that e_{i_1} starts at u, e_{i_ℓ} ends at v, the end node of each e_{i_j} is the start node of $e_{i_{j+1}}$, and the index sequence $i_1, \ldots, i_\ell$ is strictly increasing.*

We now identify a combinatorial condition that the interaction graph should satisfy in order to accommodate MPC with a given set O of parties who receive an output.

Definition 3.3 (O-connected graph). *Let G_M be an n-party interaction graph and let $O \subseteq [n]$. We say that G_M is* O-connected *if for any (not necessarily distinct) pair of nodes $s, o \in [n]$ with $o \in O$, and any node $h \in [n] \setminus \{s, o\}$, there is a trail from s to o passing through h.*

Note, in particular, that the above connectivity requirement implies the existence of a trail from every node to every output node.

We now show that the above connectivity requirement is indeed necessary to realize the standard notion of security against semi-honest adversaries. We prove this for an explicit functionality that can be thought of as a natural multi-party variant of oblivious transfer. Intuitively, this functionality has the property that the adversary only learns partial information about honest parties' inputs by invoking it once, but can learn full information by invoking it twice, on any pair of input-tuples that differ in only one entry.

Definition 3.4 (MOT functionality). *For $n \geq 2$ and nonempty $O \subseteq [n]$, let $\mathsf{MOT}_O : X^n \to Y^n$ be the n-party functionality defined as follows:*

- *The input domain of each party is $X = \{0,1\}^3$ and the output domain is $Y = \{0,1\}^{n+1}$.*

– *Given input* (c_i, x_i^0, x_i^1) *from each party* P_i*, the functionality lets* $c = c_1 \oplus \cdots \oplus c_n$ *and outputs* $(c, x_1^c, \ldots, x_n^c)$ *to all parties* P_j*,* $j \in O$ *(the output of party* P_j *for* $j \notin O$ *is the fixed string* 0^{n+1}*).*

The proof of the following lemma formalizes an argument made in [20].

Lemma 3.5. *Let* $n \geq 2$ *and* $O \subseteq [n]$ *where* $|O| \geq 1$*. Suppose* Π *securely realizes* MOT_O *in the presence of a semi-honest, static adversary who may corrupt any number of parties, where* Π *may use an arbitrary correlated randomness setup. If* Π *complies with an interaction pattern* M*, then the interaction graph* G_M *must be* O*-connected. Moreover, this holds even in the augmented semi-honest model, where the simulator can change the inputs of corrupted parties.*

Proof. The high level idea is that in the ideal model, even if the simulator can arbitrarily choose the inputs of $n-1$ corrupted parties, it can only learn one out of the last two input bits of the remaining party. We show that in the protocol, a semi-honest adversary can learn both input bits of an uncorrupted party, contradicting security. We formalize this below.

Since G_M is not O-connected, there exist nodes $s, o \in [n]$ with $o \in O$ and $h \in [n] \setminus \{s, o\}$ such that all trails from s to o avoid h. We argue that the latter implies that if all parties except h are corrupted, then by running Π *once* on inputs $x_i = 000$ for all corrupted parties P_i, $i \neq h$, and an unknown input $x_h = (c_h, x_h^0, x_h^1)$ for party P_h, the adversary can efficiently compute the entire input x_h from its view. Indeed, the adversary can recover x_h from (1) the output MOT_O delivers to party P_o on inputs $(x_1, \ldots, x_n)$, obtained directly from the honest execution; and (2) the output of MOT_O on a slightly modified input, where x_s is replaced by $x'_s = 100$. The latter output can be obtained by running a mental experiment in which the view of party P_o on the modified input is simulated given the messages sent out by party P_h in the original execution.

The simulation will simply compute the exact set of messages received by party P_o on the same local inputs and random inputs, with the only difference that $x_s = 000$ is replaced by $x'_s = 100$. To see that this is possible given the information available to the adversary, note that every message sent in the protocol can be viewed as a deterministic function of the local inputs and random inputs of the n parties. If some message received by party P_h can depend on the input of party P_s, then this message cannot influence the view of party P_o; otherwise this would imply a trail from s to o passing through h. The adversary can therefore sequentially compute whatever modified messages are implied by the information it

has (namely, inputs and random inputs of corrupted parties and messages sent out by party P_h), which includes all messages received by P_o. □

Given Lemma 3.5, it suffices to prove a lower bound on the number of edges in an O-connected interaction graph G_M. We start with the case of a single output node $O = \{o\}$ and later extend it to the general case. The proof relies on the following lemma.

Lemma 3.6. *Let $n \geq 2$ and $O = \{o\}$ where $o \in [n]$. Suppose G_M is O-connected and $v \in [n] \setminus O$ has indegree $d \geq 2$ and outdegree 1. Then there is an O-connected $G_{M'}$ with the same number of edges in which v has indegree 1 and outdegree 1.*

Proof. Let $e_{i_1}, \ldots, e_{i_d}$ be the edges entering v, where $i_1 < \cdots < i_d$. We obtain $G_{M'}$ from G_M by replacing every edge $e_{i_j} = (u_j, v)$, $1 \leq j \leq d-1$, by the edge $e'_{i_j} = (u_j, u_d)$, where u_d is the source of e_{i_d}. An example of this transformation is given in Figure 1 below. The transformation does

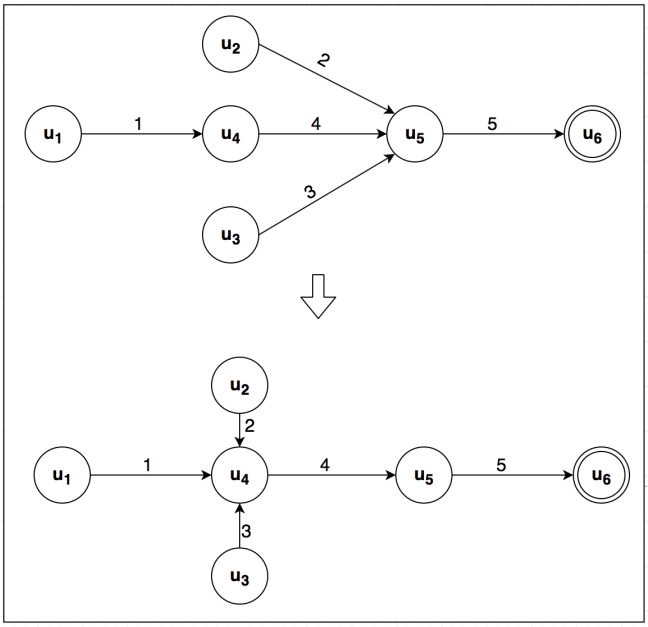

Fig. 1. Illustrating the graph transformation in the proof of Lemma 3.6. Here $o = u_6$ is the output node and $v = u_5$ is the non-output node with outdegree 1 and indegree $d = 3 \geq 2$.

not change the number of edges. It leaves e_{i_d} as the only edge entering v

and does not add outgoing edges from v, thus making both the indegree and outdegree of v equal to 1 as required. Finally, since i_d is larger than the indices of all edges e'_{i_j} whose new endpoint is u_d, any trail in G_M can be replaced by a valid trail in $G_{M'}$ with the same source and destination and with a superset of the nodes of the original trail (the new trail may replace a direct edge to v by a 2-edge sub-trail passing through u_d). This implies that $G_{M'}$ is also O-connected, as required. □

We are now ready to prove a lower bound on the number of edges for the case $|O| = 1$.

Proposition 3.7. *Let $n \geq 2$ and $O = \{o\}$ where $o \in [n]$. Suppose G_M is an O-connected n-party interaction graph. Then G_M has at least $2n - 2$ edges.*

Proof. We prove the proposition by induction on n. For the base case of $n = 2$, note that (without loss of generality) letting $s = o = 1$ and $h = 2$ imposes the existence of a trail from 1 to 1 passing through 2, which requires $m \geq 2 = 2 \cdot 2 - 2$ edges as required.

For the induction step, suppose that the proposition holds for all $k < n$, and let G_M be an O-connected n-party interaction graph with m edges. Assume towards contradiction that $m \leq 2n - 3$. We show that under this assumption, G_M can be converted into an O-connected $(n-1)$-party $G_{M'}$ that has $m' = m - 2$ edges. By the induction's hypothesis, this implies that $m' \geq 2(n-1)-2$, and so $m = m'+2 \geq 2n-2$, leading to the desired contradiction.

The transformation from G_M to $G_{M'}$ proceeds as follows. Since O-connectivity requires each node to have at least one outgoing edge, and since $m < 2n - 2$, there must be a non-output node v whose outdegree is exactly 1. (If all outdegrees are bigger than 1, then the non-output nodes alone contribute at least $2n - 2$ edges.) Moreover, the O-connectivity of G_M also requires the indegree of v to be at least 1 (e.g., letting $v = h$ and $s = o$). By Lemma 3.6, we may assume without loss of generality that the indegree of v is also 1.

Let $e_{i_1} = (u_1, v)$ be the single edge entering v and $e_{i_2} = (v, u_2)$ be the single edge existing v. Since v should be reachable, we have $i_1 < i_2$. If $u_1 = u_2$, we can obtain $G_{M'}$ by just removing v and the two incident edges from G_M. The resulting graph $G_{M'}$ has $n - 1$ nodes and $m - 2$ edges as required, and it is O-connected because every trail in G_M that passes through v has a corresponding trail in $G_{M'}$ with the same source and destination that traverses the same set of nodes excluding v.

It remains to deal with the case where $u_1 \neq u_2$. The O-connectivity of G_M implies the existence of a trail $\tau_{u_2,v,o}$ from u_2 to the output node passing through v. We obtain $G_{M'}$ from G_M by removing the node v, replacing the two edges e_{i_1}, e_{i_2} by the single edge $e'_{i_1} = (u_1, u_2)$ (with index i_1), and removing the first edge $e_{i_0} = (u_2, u_3)$ of $\tau_{u_2,v,1}$. Again, $G_{M'}$ has $n-1$ nodes and $m-2$ edges as required. Replacing e_{i_1}, e_{i_2} by e'_{i_1} clearly does not hurt O-connectivity, since (as before) any trail passing through v can be replaced by a similar trail that only excludes v. We need to show that removing the edge e_{i_0} also does not hurt O-connectivity. Note that, since $\tau_{u_2,v,o}$ should pass through e_{i_1} and then e_{i_2}, we have $i_0 \leq i_1 < i_2$. We show that for any $h \neq u_2, o$, a trail $\tau_{u_2,h,o}$ from u_2 to o via h can be replaced by a trail $\tau'_{u_2,h,o}$ in $G_{M'}$. Indeed, by the O-connectivity of G_M, there is a trail $\tau_{v,h,o}$ in G_M from v to o via h. This trail starts with e_{i_1}, and thus all of its other edges have indices bigger than i_1. Removing the first edge e_{i_1}, we get a trail $\tau'_{u_2,h,o}$ that does not use e_{i_0} (since $i_0 \leq i_1$), as required. □

Finally, we extend the lower bound of Proposition 3.7 to the case of more than one output. This relies on the following lemma.

Lemma 3.8. *Let $n \geq 2$ and $O \subseteq [n]$ be a set of $k = |O| \geq 2$ output nodes. Let M be a minimal interaction pattern such that G_M is O-connected. Then:*

1. *The last edge in M enters an output node in O;*
2. *Removing this last edge results in an interaction pattern M' such that $G_{M'}$ is O'-connected for some $O' \subset O$ with $|O'| = |O| - 1$.*

Proof. If the last edge in M does not enter an output node from O, then it can be removed from M without hurting the O-connectivity of G_M, contradicting minimality. Now suppose that the last edge in M enters $o \in O$. Removing this last edge from G_M results in an O' interaction graph for $O' = O \setminus \{o\}$. Indeed, since the removed edge has a maximal index, it cannot be used as an intermediate edge in any trail ending in $o' \in O'$. □

Combining Proposition 3.7 and Lemma 3.8 we get the main theorem of this section.

Theorem 3.9. *Let $n \geq 2$ and $O \subseteq [n]$ be a set of $k = |O| \geq 1$ output nodes. Suppose G_M is an O-connected n-party interaction graph. Then G_M has at least $2n + k - 3$ edges.*

Proof. The theorem follows by induction on k, using Proposition 3.7 as the base case ($k = 1$) and Lemma 3.8 for the induction step. □

Together with Lemma 3.5, we get the following corollary:

Corollary 3.10. *Let $n \geq 2$ and $O \subseteq [n]$ where $|O| = k \geq 1$. Suppose Π securely realizes MOT_O in the presence of a semi-honest, static adversary who may corrupt any number of parties, where Π may use an arbitrary correlated randomness setup. If Π complies with an interaction pattern M, then M involves at least $2n+k-3$ messages. Moreover, this holds even in the augmented semi-honest model, where the simulator can change the inputs of corrupted parties.*

4 Upper Bounds

In this section we complement the lower bound from Section 3 by presenting matching upper bounds in several different models. We note that our focus here is on the computational model of security, which allows us to bypass strong lower bounds for the information-theoretic model from the recent work of Damgård et al. [13].

Using standard general transformations (cf. [18]), the secure computation of any (non-reactive) randomized multi-output functionality f can be reduced to the secure computation of a related deterministic, functionality f' that delivers the same output to all parties. This reduction does not incur additional messages. We thus restrict our attention to the latter type of functionalities.

As a final simplification, it suffices to prove an upper bound of $2n - 2$ messages for the case only one party has an output. Indeed, in the case of $k > 1$ parties should receive the output, we can first deliver the output to one of these parties using $2n - 2$ messages, and then use $k - 1$ additional messages to communicate the output to the other parties. This yields a total of $2n + k - 3$ messages, as required.

Theorem 4.1. *Let f be an n-party functionality delivering output to party P_1. Suppose there is a 2-round n-party MPC protocol Π for f in the common random string (CRS) model. Then there is a similar protocol Π' for f in the plain model in which the parties send a total of $2n - 2$ point-to-point messages. Furthermore, if Π relies on a trusted source of correlated random inputs, then Π' can be implemented using the same correlated randomness.*

Proof. We assume for simplicity that Π does not rely on correlated randomness other than (possibly) a CRS. The "furthermore" part of the theorem is obtained by a straightforward extension of the following proof.

Let $\alpha_{i,j}$ denote the message sent from P_i to P_j in Round 1, and β_i the message sent from P_i to P_1 in Round 2. The high level idea is to use a "two-way chain" interaction pattern moving from P_1 to P_n and back to P_1, where at each point each party computes whatever messages it can given the information received so far and forwards these messages along with previous information it received to the next party. Concretely, protocol Π' emulates the messages of Π as follows:

1. P_1 picks the CRS σ, and based on σ, its local input, and its local randomness computes the messages $\alpha_{1,j}$ for all $2 \leq j \leq n$. It sends a single message consisting of σ and the $n-1$ messages $\alpha_{1,j}$ to P_2.
2. For $i = 2, \ldots, n-1$, party P_i uses the Π'-message α'_{i-1} received from P_{i-1} to compute the Π-messages $\alpha_{i,j}$, for all $j \neq i$, and sends these messages to P_{i+1} together with the information received from P_{i-1}.
3. Party P_n uses the CRS σ, its local input, and its local randomness to compute the messages $\alpha_{n,j}$, $1 \leq j \leq n-1$. It additionally uses the messages $\alpha_{i,n}$ received from P_{n-1} to compute the message β_n. It sends the messages $\alpha_{n,j}$ and β_n to P_{n-1} along with the message of P_{n-1}.
4. For $i = n-1, \ldots, 2$, party P_i uses its local input, local randomness, and the information received from P_{i+1} to compute the message β_i. It sends β_i along with the message it received from P_{i+1} to P_{i-1}.
5. Party P_1 uses its local input, local randomness, and the information received from P_2 to compute the output of Π.

Overall, the protocol involves $2n-2$ messages ($n-1$ in each direction), as required. Correctness follows from the fact that Π' perfectly emulates the messages sent in Π. Security follows from the fact that the view of any (static, semi-honest) adversary corrupting a subset of the parties in Π' is identically distributed (up to message ordering) to the view of a similar adversary corrupting the same subset of parties in Π. □

Using recent 2-round MPC protocols from [4, 17], we get the following corollary for message-optimal MPC in the plain model.

Corollary 4.2. *Suppose a 2-message (semi-honest) oblivious transfer protocol exists. Then, any polynomial-time n-party functionality delivering output to k parties can be securely computed in the plain model with $2n + k - 3$ messages.*

We note that the assumption that a 2-message oblivious transfer protocol exists is necessary, since such a protocol is a special case of Corollary 4.2 with $n = 2$ and $k = 1$.

We are able to further reduce the computational assumptions in the offline-online model, where a trusted source of (input-independent) correlated randomness is available. The latter can be generated by the parties themselves using an interactive MPC protocol that is carried out in an offline, input-independent preprocessing phase. Given a correlated randomness setup, 2-round MPC becomes considerably easier [10, 21]. In particular, such protocols can be achieved unoconditionally for functionalities in low complexity classes such as NC^1, or can be based on any one-way function for general polynomial-time computable functionalities. The following theorem is informally mentioned in [21], we provide a proof sketch for self-containment.

Theorem 4.3. *Suppose a one-way function exists. Then, any polynomial time n-party functionality f can be realized by a 2-round protocol with a correlated randomness setup. Furthermore, the same result holds unconditionally (and with information-theoretic security) for functionalities f in the complexity class* NC^1 *or even (uniform)* NL/poly.

Proof. (sketch) Assume for simplicity that each input of f is a single bit and the output is only revealed to P_1; the general case can be reduced to this case. Consider any decomposable randomized encoding [15, 1, 22] (or projective garbling [2]) for f. Such an encoding can be expressed as an efficiently samplable joint distribution $R_f = ((r_1^0, r_1^1), \ldots, (r_n^0, r_n^1))$ such that given $(r_{x_1}^1, \ldots, r_{x_n}^n)$ one can recover $f(x)$ but cannot learn anything else about x. The existence of such R_f for polynomial-time computable functionalities f (with computational hiding of x) can be based on any one-way function [24]. For functions f in NC^1 or even (uniform) NL/poly, it exists unconditionally with perfect hiding of x [15, 1].

Given R_f as above, a protocol for f in the correlated randomness model proceeds as follows. To generate the correlated randomness, sample $((r_1^0, r_1^1), \ldots, (r_n^0, r_n^1))$ from R_f, pick a secret mask $\rho_i \in \{0, 1\}$ for each input bit x_i, and use n-out-of-n (e.g., additive) secret sharing to share each (r_i^0, r_i^1) between the parties, where the pair entries are permuted according to ρ_i. That is, each party gets a "left share" of $r_i^{\rho_i}$ and a "right share" of $r_i^{1-\rho_i}$. Moreover, the permutation bit ρ_i is revealed to party P_i.

In the online phase, on input x_i, party P_i sends its masked input $x_i' = x_i \oplus \rho_i$ to all other parties. In the second round, each party sends to P_1 the n shares corresponding to the bits x_i', namely if $x_i' = 0$ then the

left share (of $r_i^{\rho_i}$) is sent and otherwise the right share (of $r_i^{1-\rho_i}$) is sent. Given the shares received from all parties, P_1 reconstructs $(r_{x_1}^1, \ldots, r_{x_n}^n)$, from which it can decode $f(x_1, \ldots, x_n)$. Security follows from the security of the randomized encoding and the fact that the unrevealed values $r_{1-x_i}^i$ are not revealed to the adversary even when corrupting an arbitrary strict subset of the parties. □

Combining Theorem 4.1 and Theorem 4.3, we get the following corollary for message-optimal MPC with correlated randomness setup.

Corollary 4.4. *Suppose a one-way function exists. Then, any polynomial time n-party functionality f delivering output to k parties can be securely computed with a correlated randomness setup and $2n + k - 3$ online messages. Furthermore, the same result holds unconditionally (and with information-theoretic security) for functionalities f in the complexity class* NC^1 *or even (uniform)* NL/poly.

5 Conclusions and Future Research

In this work we provide a tight characterization of the message complexity of computationally secure MPC in the presence of semi-honest adversaries that can corrupt any number of parties. Our work leaves several natural directions for future research.

One direction is understanding the type of achievable security and necessary setup for extending the positive results to accommodate malicious adversaries. While such an extension is fairly simple in some settings (e.g., for NC^1 functions with a correlated randomness setup and settling for "security with selective abort" [21]), characterizing the minimal message complexity in the plain model or with stronger forms of security seems like a challenging problem.

Another direction is to better understand the message complexity of MPC in the case where at most t parties can be corrupted. This relaxed setting is more sensitive to the distinction between static vs. adaptive corruption (with or without erasures) and between fixed vs. dynamic interaction pattern. Partial results are given in [11, 7, 8, 5, 13, 16].

Acknowledgements. The first and third authors were supported in part by NSF-BSF grant 2015782 and BSF grant 2012366. The first author was additionally supported by ERC grant 742754, ISF grant 1709/14, DARPA/ARL SAFEWARE award, NSF Frontier Award 1413955, NSF grants 1619348, 1228984, 1136174, and 1065276, a Xerox Faculty Research Award, a Google Faculty Research Award, an equipment grant

from Intel, and an Okawa Foundation Research Grant. This material is based upon work supported by the DARPA through the ARL under Contract W911NF-15-C-0205. The third author was additionally supported by NSF grant 1619348, DARPA, OKAWA Foundation Research Award, IBM Faculty Research Award, Xerox Faculty Research Award, B. John Garrick Foundation Award, Teradata Research Award, and Lockheed-Martin Corporation Research Award. The views expressed are those of the authors and do not reflect the official policy or position of the DoD, the NSF, or the U.S. Government.

References

1. B. Applebaum, Y. Ishai, and E. Kushilevitz. Cryptography in NC^0. In *FOCS*, pages 166–175, 2004.
2. M. Bellare, V. T. Hoang, and P. Rogaway. Foundations of garbled circuits. In *Proc. CCS '12, Raleigh, NC, USA, October 16-18, 2012*, pages 784–796, 2012.
3. M. Ben-Or, S. Goldwasser, and A. Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract). In *STOC*, pages 1–10, 1988.
4. F. Benhamouda and H. Lin. k-round MPC from k-round ot via garbled interactive circuits. Cryptology ePrint Archive, Report 2017/1125, 2017. https://eprint.iacr.org/2017/1125.
5. E. Boyle, K. Chung, and R. Pass. Large-scale secure computation: Multi-party computation for (parallel) RAM programs. In *Proc. CRYPTO 2015, Part II*, pages 742–762, 2015.
6. E. Boyle, N. Gilboa, Y. Ishai, H. Lin, and S. Tessaro. Foundations of homomorphic secret sharing. In *Proceedings of ITCS 2018*, pages 21:1–21:21, 2018. Full version: https://eprint.iacr.org/2017/1248.
7. E. Boyle, S. Goldwasser, and S. Tessaro. Communication locality in secure multi-party computation - how to run sublinear algorithms in a distributed setting. In *Proceedings of TCC 2013*, pages 356–376, 2013.
8. N. Chandran, W. Chongchitmate, J. A. Garay, S. Goldwasser, R. Ostrovsky, and V. Zikas. The hidden graph model: Communication locality and optimal resiliency with adaptive faults. In *Proceedings ITCS 2015*, pages 153–162, 2015.
9. D. Chaum, C. Crépeau, and I. Damgård. Multiparty unconditionally secure protocols (extended abstract). In *STOC*, pages 11–19, 1988.
10. S. G. Choi, A. Elbaz, T. Malkin, and M. Yung. Secure multi-party computation minimizing online rounds. In *Proc. ASIACRYPT 2009*, pages 268–286, 2009.
11. B. Chor and E. Kushilevitz. A communication-privacy tradeoff for modular addition. *Inf. Process. Lett.*, 45(4):205–210, 1993.
12. R. Cramer, I. Damgård, and J. B. Nielsen. Multiparty computation from threshold homomorphic encryption. In *Proc. EUROCRYPT 2001*, pages 280–299, 2001.
13. I. Damgård, J. B. Nielsen, R. Ostrovsky, and A. Rosén. Unconditionally secure computation with reduced interaction. In *Proc. EUROCRYPT 2016, Part II*, pages 420–447, 2016.
14. I. Damgård, J. B. Nielsen, A. Polychroniadou, and M. A. Raskin. On the communication required for unconditionally secure multiplication. In *Proc. CRYPTO 2016, Part II*, pages 459–488, 2016.

15. U. Feige, J. Killian, and M. Naor. A minimal model for secure computation (extended abstract). In *Proceedings of the Twenty-sixth Annual ACM Symposium on Theory of Computing*, STOC '94, pages 554–563, New York, NY, USA, 1994. ACM.
16. J. A. Garay, Y. Ishai, R. Ostrovsky, and V. Zikas. The price of low communication in secure multi-party computation. In *Proc. CRYPTO 2017, Part I*, pages 420–446, 2017.
17. S. Garg and A. Srinivasan. Two-round multiparty secure computation from minimal assumptions. Cryptology ePrint Archive, Report 2017/1156, 2017. https://eprint.iacr.org/2017/1156.
18. O. Goldreich. *The Foundations of Cryptography - Volume 2, Basic Applications*. Cambridge University Press, 2004.
19. O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game or a completeness theorem for protocols with honest majority. In *STOC*, pages 218–229, 1987.
20. S. Halevi, Y. Ishai, A. Jain, E. Kushilevitz, and T. Rabin. Secure multiparty computation with general interaction patterns. In *Proc. ITCS 2016*, pages 157–168, 2016. Full version: https://eprint.iacr.org/2015/1173.pdf.
21. Y. Ishai, E. Kushilevitz, S. Meldgaard, C. Orlandi, and A. Paskin-Cherniavsky. On the power of correlated randomness in secure computation. In *Proc. TCC 2013*, pages 600–620, 2013.
22. Y. Ishai, E. Kushilevitz, R. Ostrovsky, and A. Sahai. Cryptography with constant computational overhead. In *STOC*, pages 433–442, 2008.
23. P. Mukherjee and D. Wichs. Two round multiparty computation via multi-key FHE. In *Proc. EUROCRYPT 2016, Part II*, pages 735–763, 2016.
24. A. C.-C. Yao. How to generate and exchange secrets (extended abstract). In *FOCS*, pages 162–167, 1986.

A Secure Multiparty Computation

For completeness, we provide here an overview of the standard definition of MPC we use. We refer the reader to [18] for a more complete treatment.

We consider by default an n-party *functionality* f to be a deterministic mapping of n inputs to n outputs.

An n-party *protocol* Π prescribes a randomized interaction between parties $P_1, \ldots, P_n$ on their local inputs x_i. This interaction may proceed in rounds, where in each round each party can send a message to each other party. Since our current focus on message complexity rather than round complexity, we may assume without loss of generality that only a single message is sent in each round. Formally, Π is a polynomial-time computable next message function that on input i (party identity), 1^k (global security parameter), x_i (local input of P_i), r_i (local random input of P_i) and (m_{i_j}) (sequence of messages received so far by P_i) specifies the next message P_i should send and its destination, or alternatively the local

output y_i of P_i. In the plain model, the r_i are independently random bit-strings, whereas in the correlated randomness model they can be picked by a PPT sampling algorithm $D(1^k)$.

We make the following correctness requirement: if parties $P_1, \ldots, P_n$ interact according to Π on inputs 1^k and $(x_1, \ldots, x_n)$, then they end up with local outputs $(y_1, \ldots, y_n) = f(x_1, \ldots, x_n)$ except with negligible probability in k.

The security of a protocol (with respect to the functionality f) is defined by comparing the real-world execution of the protocol with an ideal-world evaluation of f by a trusted party. More concretely, it is required that for every adversary Adv, which attacks the real execution of the protocol, there exist an adversary Sim, also referred to as a simulator, which can *learn essentially the same information* in the ideal-world. Since we consider security against semi-honest adversaries and deterministic functionalities, we are only concerned with simulating the view of Adv and not its effect on the outputs of uncorrupted parties.

The real execution. In the real execution of Π, the adversary Adv, given an auxiliary input z, corrupts a set $I \subset [n]$ of the parties and outputs their entire view. This view consists (without loss of generality) of their inputs x_i, random inputs r_i, and messages received from other parties. (The outgoing messages are determined by the above information.) The output of Adv on a protocol Π defines a random variable $\mathsf{REAL}_{\pi,\mathsf{Adv}(z),I}(k, \boldsymbol{x})$.

The ideal execution. In the ideal world, there is a trusted party who computes f on behalf of the parties. The simulator Sim, given an auxiliary input z, corrupts a set $I \subset [n]$, receives the inputs and outputs of parties in I, and computes some (randomized) function of this information. The interaction of Sim with f defines a random variable $\mathsf{IDEAL}_{f,\mathsf{Sim}(z),I}(k, \boldsymbol{x})$ whose value is determined by the random coins of Sim.

Having defined the real and the ideal executions, we now proceed to define our notion of security. We say that Π securely computes f in the presence of semi-honest adversaries if for every $I \subset [n]$ and PPT adversary Adv (whose running time is polynomial in k) there exists a PPT simulator Sim, such that for every sequence of polynomial-size auxiliary inputs z_k and inputs $\boldsymbol{x} = (x_1, \ldots, x_n)$, the following quantity is negligible in k:

$$|\Pr[\mathsf{REAL}_{\Pi,\mathsf{Adv}(z),I}(k, \boldsymbol{x}) = 1] - \Pr[\mathsf{IDEAL}_{f,\mathsf{Sim}(z),I}(k, \boldsymbol{x}) = 1]|.$$

We also consider the case of *information-theoretic security*, in which both Adv and Sim are computationally unbounded.