Systematizing Secure Computation for Research and Decision Support

Jason Perry[1], Debayan Gupta[2], Joan Feigenbaum[2]
and Rebecca N. Wright[1]

[1] Rutgers University, NJ, USA. {jason.perry|rebecca.wright}@rutgers.edu
[2] Yale University, CT, USA. {debayan.gupta|joan.feigenbaum}@yale.edu

Abstract. We propose a framework for organizing and classifying research results in the active field of secure multiparty computation (MPC). Our *systematization of secure computation* consists of (1) a set of definitions circumscribing the MPC protocols to be considered; (2) a set of quantitative axes for classifying and comparing MPC protocols; and (3) a knowledge base of propositions specifying the known relations between axis values. We have classified a large number of MPC protocols on these axes and developed an interactive tool for exploring the problem space of secure computation. We also give examples of how this systematization can be put to use to foster new research and the adoption of MPC for real-world problems.

1 Introduction

For more than 30 years, since the groundbreaking work of Yao [30,31] and Goldreich *et al.* [18], hundreds of research papers on *Secure Multiparty Computation* (MPC) have appeared, many of them proposing original protocols for carrying out general secure computation under varying sets of assumptions. In this paper, we systematically organize the main research results in this area, in order to:

- Help potential users of MPC learn which existing protocols and implementations best match the sensitive-data computations they would like to perform. This may stimulate adoption of MPC in areas where it would be beneficial.
- Help new researchers get up to speed in a complex area by providing an overview of the "lay of the land."
- Help MPC researchers explore the problem space and discover remaining openings for protocols with new combinations of requirements and security features—or for new impossibility results that preclude the existence of such protocols.

Most research papers in MPC include comparisons of their results to related work, often with tables related to the most significant protocol features, in order to provide context for understanding the paper's contributions. However, none has attempted to organize the larger problem space in order to meet the goals listed above. There are a handful of introductory surveys and textbook-like

treatments of MPC [17,11,23,12]; these have (justifiably) focused on a narrower region of the problem space or specific security model, in order to present the material in a pedagogically clean way. In contrast, we do not limit ourselves to one model or set of definitions but instead provide a framework for examining their variations.

Our work shares some common goals with research meant to foster the real-world adoption of secure MPC. Such papers include the MPC-in-the-field experiments of Feigenbaum *et al.* and Bogetoft *et al.* [15,8] and the end-user survey work of Kamm *et al.* [7].

Since an effort such as this can never comprehensively account for every piece of research in a sprawling and active field, the framework has been deliberately designed to be extensible; it can accommodate new results and refined definitions without breaking.

In Section 2, we present the three major components that we believe are needed to systematize the main body of MPC results: (1) a set of definitions delineating the boundaries of the problem space; (2) a set of quantitative features for describing protocols; and (3) a knowledge base of propositions specifying the known relationships and dependencies among features. In Section 3, we describe the construction of a systematization database of more than 60 significant MPC protocols, and in Section 4 we present a user interface designed to aid the exploration of the database of systematized MPC protocols and show how our systematization can be put to work to facilitate new research.

2 The Systematization

As a necessary prerequisite for this work, we have carried out an extensive literature survey producing an annotated bibliography of MPC research. It contains over 180 papers from the MPC literature, as well as a sampling of important papers for related problems, including secret sharing and oblivious transfer.

In addition to a written paragraph annotating each paper, the BibTeX source of our Annotated Bibliography includes tags for each paper indicating which aspects of MPC it treats, *e.g.*, `2party` for a paper with a specifically two-party protocol, or `uncond` for a protocol with unconditional security. These tags make it possible to write scripts to automatically generate a bibliography for any specific sub-problem or aspect of MPC.

The bibliography continues to be updated on an ongoing basis. The most recent version is available online [27].

2.1 Definitions

In this section, we provide definitions to delimit the scope of the secure computation protocols that we are concerned with. These definitions are purposely quite broad, so that a large range of work can potentially be captured by them.

Variables that determine the fundamental nature of an MPC protocol include: (1) whether the protocol is for a fixed number of parties (most commonly,

two) or is for any number $n \geq 2$ of parties, (2) whether the protocol is for computing one specific functionality (*e.g.*, set intersection) or any of a class of functionalities, and (3) whether the protocol treats exact computation only or secure computation of approximations, which is a generalization of exact MPC [14]. Our initial literature survey encompassed all of these, although not exhaustively. For the current systematization, we consider only protocols for exact computation, and thus we have four definitions, one for each setting of the two variables.

Since the way that security is defined varies from protocol to protocol, and indeed a primary purpose of our systematization is to examine such variations, our definitions necessarily cannot give any fixed definition of security. What matters is that, in the literature, protocols are proven to meet rigorous definitions of security and that our systematization indicates which definitions are actually in use for a given protocol. Therefore, for an MPC protocol to be considered as a candidate for systematization, in addition to fitting into one the definitions listed below, it must be accompanied by rigorous definitions of security, including privacy of inputs and correctness of outputs, that the protocol has been proven to meet. The nature of these security definitions is elaborated in the next subsection.

Definition 1. *A protocol for* Secure n-party Computation of a functionality f *is a specification of an interactive process by which a fixed number n of players, each holding a private input x_i, can compute a specific, possibly randomized, functionality of those inputs $f(x_1, ..., x_n) = (y_1, ..., y_n)$.*

Definition 2. *A protocol for* Secure Multiparty Computation of a functionality f *is a specification of an interactive process by which any number $n \geq 2$ of players, each holding a private input x_i, can compute a specific, possibly randomized, functionality of those inputs $f(x_1, ..., x_n) = (y_1, ..., y_n)$.*

Definition 3. *A protocol for* Secure n-party Computation of a class $\mathcal{C}$ of functionalities *allows a fixed number n of players, each holding a private input x_i, to compute an agreed-upon, possibly randomized, functionality of those inputs $f(x_1, ..., x_n) = (y_1, ..., y_n)$, where f is any member of the class $\mathcal{C}$ of functionalities.*

Definition 4. *A protocol for* Secure Multiparty Computation of a class $\mathcal{C}$ of functionalities *allows any number $n \geq 2$ of players, each holding a private input x_i, to compute an agreed-upon, possibly randomized, functionality of those inputs $f(x_1, ..., x_n) = (y_1, ..., y_n)$, where f is any member of the class $\mathcal{C}$ of functionalities.*

The class $\mathcal{C}$ is typically used to refer to the model of computation in which a protocol's functionalities are represented, such as circuits or RAM programs. A majority of the work in MPC has been concerned with universal (Turing-complete) computation, but there has been work exploring secure computation specifically for restricted computation classes, such as AC0 or NC1 circuits or regular or context-free languages.

All of the protocols we surveyed for the systematization fall under one of these definitions, with the majority coming under the most general definition (Definition 4). Yao-like two-party computation protocols fall under Definition 3.

2.2 Linear axis representation of MPC protocol features

The main conceptual object in our systematization is a set of linear *axes*, where each axis represents an ordering of values of a single feature of MPC protocols. Every axis has at least two labeled values, at the endpoints. Some axes are continuous and others discrete. MPC protocols can be scored on these axes, allowing them to be compared quantitatively. This is the first attempt we are aware of to factor research results in MPC into such a representation. The axes were selected based on our literature survey, using two guiding principles:

1. The axes should be as orthogonal as possible, minimizing overlap (although some logical dependencies between axes are unavoidable.)
2. The set of axes should be *complete* in the sense that they can express all distinctions of security and (asymptotic) efficiency between any two protocols.

For any discrete axis that is not inherently binary, the number of occupied intermediate values on the axis is subject to change. The diagrams below show intermediate values that are known to have been achieved by MPC protocols. However, this should not be seen as finally determining the number of points on the axis. Indeed, one of the objectives of the axis representation is to highlight the possibility of future protocols with new intermediate values. This has already happened for several axes over the history of MPC research. For example, the appearance of protocols tailored for covert adversaries, such as those of Aumann *et al.* and Goyal *et al.* [1,22], showed that there are intermediate values along the "Maliciousness" axis (Axis 7), whereas previously only the passive/malicious distinction had been considered.

We orient all the axes in the same direction, such that moving from left to right on a given axis indicates an improved protocol—*e.g.*, one that is more efficient, has a stronger security guarantee, or requires a weaker setup or computational-hardness assumption. In drawing the non-numeric axes, the points have been placed with equal spacing; the relative distances between points on these axes should not be considered significant.

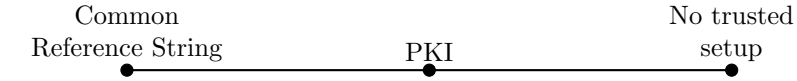
Our axes do not include the model of computation in which a protocol is expressed. This is a categorical feature which is indicated by the definition (Section 2.1) under which the protocol falls. The model of computation for each protocol is included in its entry in our MPC protocol database, described in Section 3. We have not generated axes for proof techniques, because a proof technique is not a function of the protocol; a protocol's security may be proven in a number of ways.

We now proceed to describe the axes and values in detail. The axes are informally grouped into four categories, which serves to highlight the tradeoffs inherent in achieving secure multiparty computation. The axes in categories I

and II, Environmental features and Assumptions, can be thought of as what one "pays" to enable secure MPC, and categories III (Security) and IV (Efficiency) can be seen as what one is "buying." A similar tradeoff structure can also be seen on a smaller scale among axes within the efficiency category. When it is helpful, our description of an axis also cites particular MPC solutions that instantiate points on the axis.

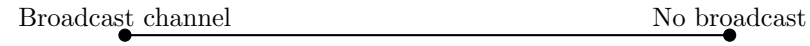
I. Environmental Features Axes
This is the category of features assumed to be provided by the execution environment. The right endpoint of these axes indicates that the feature is not required in any form.

1. Trusted setup



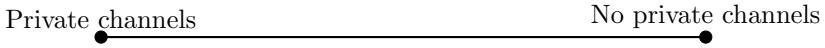
Protocols achieving the highest composable security levels require some type of trusted data to be shared by all the parties prior to the protocol execution. The middle point, PKI, is occupied by protocols such as that of Barak *et al.* [2], who showed how to use public-key-like assumptions instead of a polynomial-length common reference string in any case where computational security suffices.

2. Broadcast



The broadcast-channel assumption means that each party has the ability to send a message to all other parties simultaneously, and that all parties receiving the broadcast have assurance that the same message was received by all parties.

3. Private channels



The private channel assumption is only significant for unconditionally secure protocols, because cryptography using basic computational-hardness assumptions can be used to emulate private and authenticated communication channels.

4. Synchronization

A basic assumption of the early MPC protocols is that they operate on a *synchronous* network, in which all sent messages arrive on time and in order. The asynchronous case was first considered in Ben-Or *et al.* [4], where messages may be arbitrarily delayed and arrive in any order. Note that, in such a case, it is impossible to know whether a corrupted party has failed to send a message or, rather, the message is simply delayed. Later works, such as that of Damgård *et al.* [13], have staked

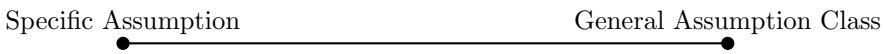
out intermediate points on this axis by giving protocols that require a smaller number of synchronization points (typically a single one).

II. Assumption Axes

5. Assumption level

A total (linear) ordering for cryptographic assumptions is not known, and the separation of assumptions cannot currently be unconditionally proven. We therefore use broader categories of assumption type, because these are usually sufficient to distinguish protocols. If a protocol makes no such assumptions, it is said to have *unconditional* security (see Axis 10). Some work specifies protocols in a *hybrid* model, with no concrete computational assumptions, but in which some high-level cryptographic operation (such as oblivious transfer) is assumed to exist as a black box. See Section 3 for a discussion of how such protocols are treated in this systematization.

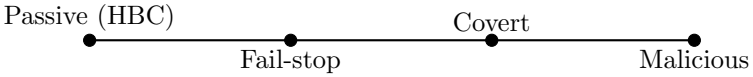
6. Specific or general assumption



Some more efficient protocols have been designed by making use of specific number-theoretic assumptions. This axis indicates whether the protocol requires such assumptions or whether it is stated so as to use any assumption from a given class, *e.g.*, trapdoor permutations.

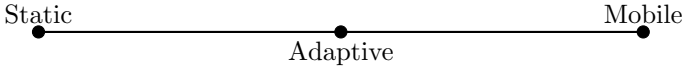
III. Security Axes

7. Adversary maliciousness



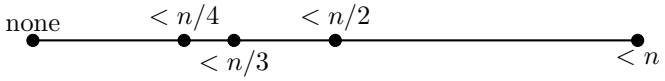
A passive, or honest-but-curious, adversary is one that follows the protocol but may use the data of corrupted parties to attempt to break the protocol's privacy. A fail-stop adversary follows the protocol except for the possibility of aborting. A malicious adversary is one whose behavior is arbitrary, and a covert adversary is like a malicious one, except that it only deviates from the protocol if the probability of being caught is low. Not present on the axis is the value "rational," since the class of rational adversaries is in fact a generalization that can encompass the entire axis, except for fully malicious, because malicious behavior can be truly arbitrary. The position of a rational adversary on the axis is determined by its utility function.

8. Adversary mobility



A static adversary must choose which parties to corrupt before the protocol begins. An adaptive adversary can choose which parties to corrupt, up to the security threshold (see Axis 9), over the course of the computation, after observing the state of previously corrupted parties. A mobile adversary is able to move corruptions from one party to another in the course of the computation.

9. Number of corrupted parties tolerated

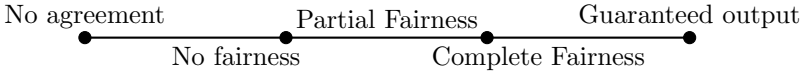


This is the maximum number of corrupted parties for which the (strongest) security guarantees of the protocol hold. The values shown are chosen merely to be representative of the most well known protocols; any value along the axis is possible.
Some protocols tolerate additional corrupted parties at a lower level of maliciousness; see axes 13 and 14.

10. Security type

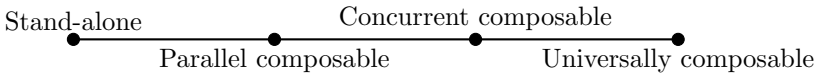
Both statistical and perfect security are unconditional, that is, not based on computational hardness assumptions. Note that true unconditional security typically cannot be achieved through the internet, even if an unconditionally-secure protocol is used, since all unconditionally secure protocols require the assumption of either private or broadcast channels, which on the internet must be emulated by cryptography.

11. Fairness guarantee



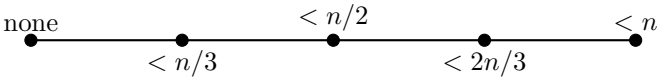
A protocol is *fair* if all honest parties receive the output if any party does. Agreement means that either all honest parties receive the output or none of them do. Protocols without agreement were introduced by Goldwasser *et al.* [20]. Some authors use the term "with abort" to refer to the no-fairness situation, in which dishonest parties can abort after receiving the correct output.

12. Composability



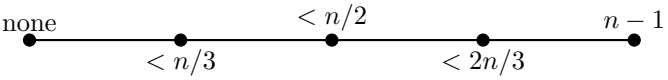
The composability guarantees of a protocol indicate whether that protocol remains secure when executed in an environment where other protocols may be executed sequentially or in parallel. The strongest guarantee, *universal composability* (UC), implies that the security properties of a protocol hold regardless of the environment in which it is executed.

13. Bound for additional passively corrupted parties tolerated



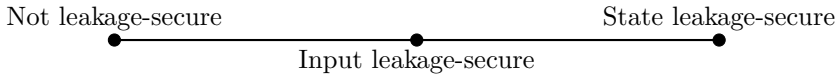
This axis applies to protocols achieving "mixed adversary" security. A protocol that tolerates a certain proportion of maliciously/covertly corrupted parties may also tolerate an additional number of passively corrupted parties, up to a certain threshold. The values on this axis represent that upper threshold. This and the following axis relate to MPC protocols with "graceful degradation", which is surveyed in Hirt *et al.* [24].

14. Corrupted parties tolerated with weakened security



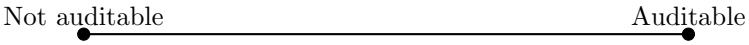
This axis applies to protocols with "hybrid security" results. A protocol that tolerates a certain proportion of corrupted parties (Axis 9) may in fact tolerate a larger number of corruptions, but with a weaker security type, *e.g.*, computational vs. unconditional security.

15. Leakage Security



Leakage security is an additional guarantee that an adversary cannot gain an advantage even if it can force all honest players to "leak" some bits of information about their state in the course of the computation. See definitions in Bitansky *et al.* [6].

16. Auditability

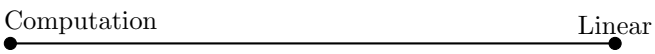


This axis indicates whether the protocol includes computations that allow for examining the transcript of computation after it is finished, to prove that the parties have correctly followed the protocol. This may be the most recent axis to come into existence, starting with the work of Baum *et al.* [3]

IV. Efficiency Axes

Our efficiency axes are concerned primarily with the asymptotic efficiency of the protocol in question.

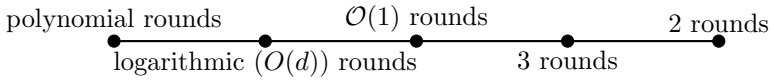
17. Online computational overhead



Historically, the main efficiency concern in MPC has been with communication rather than computational complexity; thus the lack of elaboration of this axis. More recently, Ishai *et al.* have notably shown how to

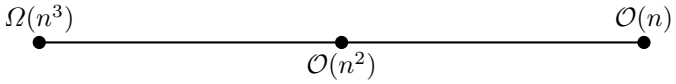
achieve MPC with constant computational overhead [25], and the RAM-model results of Gordon *et. al* [21] have shown the possibility, in the RAM model, of MPC with amortized computation that is sublinear in the input size.

18. Online communication complexity (rounds)



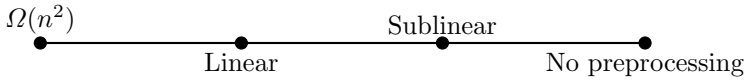
Here, d is the depth of the circuit representing the functionality. Minimizing the number of rounds of computation, independently of the total amount of bytes communicated, is crucial for efficiency in a high-latency or asynchronous network environment. Fully general MPC was shown to require at least three rounds in Gennaro *et al.* [16], although for some functionalities a 2-round protocol is possible.

19. Online communication complexity (per-gate)



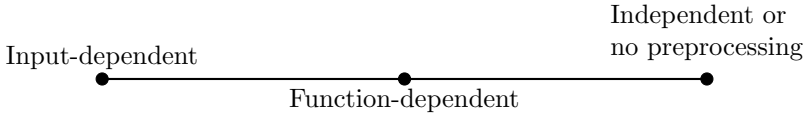
This is the most significant measure of efficiency for MPC protocols. It can represent either bits or field elements of communication. The original BGW protocol has a communication complexity of $O(n^6)$ bits per multiplication gate in the worst case. Anything cubic or worse occupies the lowest position on our axis, as finer distinctions at that level would have little value for distinguishing current, more practical protocols.

20. Preprocessing Communication complexity



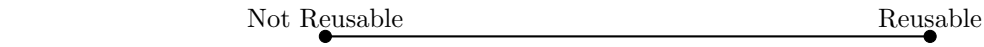
Many recent protocols achieve improved online communication efficiency by means of a preprocessing phase. In the case where the functionality is represented as an arithmetic circuit, the preprocessing phase is typically a simulation of a trusted dealer that distributes *multiplication triples*, which allow local evaluation of multiplication gates in the online phase. Sublinear preprocessing typically indicates that the preprocessing consists only of exchange of public keys, which is also indicated as a setup assumption (Axis 1).

21. Preprocessing Dependency



In some protocols, the preprocessing phase depends on the specific functionality to be computed, while in others it only depends on the upper bound of the size of the circuit. In all cases, the preprocessing is independent of the parties' inputs.

22. Preprocessing Reuse



This indicates whether the information computed in the preprocessing stage, of whatever type or amount, can be reused for multiple computations. Data of the nature of a public key typically can be reused, while *e.g.,* garbled circuits traditionally cannot be reused without breaking security. But see recent work of Goldwasser *et al.* [19].

As mentioned above, we have endeavored to make this selection of axes as complete as possible. The value of completeness in a systematization can be illustrated as follows: Suppose there are two MPC protocols whose scores along the axes are identical for every axis except one. If the set of axes is complete, then we can be confident that the protocol with the higher value on that axis is strictly better.

2.3 Dependencies Between Axes

Since many MPC protocols involve essential tradeoffs in order to achieve security or efficiency, a systematization of secure computation also needs to model what is known about how features of protocols interact. In this section, we present the second major aspect of our systematization: a list of each of the theorems known to imply constraints among the axes' values, each accompanied by a statement of the constraint. References are given to the paper in which the theorem implying the constraint was proven.

Theorem 1 ([5]). *If statistical or perfect security is obtained, then either a broadcast channel or private channels must be assumed.* ***Axis constraint****: If Axis 10's value is to the right of "Computational," then either Axis 3's value is "Private channel" or Axis 2's value is "Broadcast channel".*

Theorem 2 ([29]). *No protocol with security against malicious adversaries can tolerate more than $n/2$ corrupted parties without losing the complete fairness property.* ***Axis constraint****: If Axis 7's value is "Malicious" and Axis 9's value is to the right of $n/2$, then Axis 11's value must be to the left of "Complete fairness".*

Theorem 3 ([10]). *No protocol unconditionally secure against malicious adversaries can guarantee output delivery with $n/3$ or more corrupted parties.* ***Axis constraint****: If Axis 7's value is "Malicious," Axis 10's value is to the right of "Computational," and Axis 9's value is to the right of "$n/3$," then Axis 11's value must be to the left of "Guaranteed output".*

Theorem 4 ([5]). *No protocol can have perfect security against more than $n/3$ maliciously corrupted adversaries.* ***Axis constraint****: If Axis 7's value is "Malicious" and Axis 9's value is to the right of $n/3$, then Axis 10's value must be to the left of "Perfect".*

Theorem 5 ([16]). *Any general MPC protocol with complete fairness against a malicious adversary must have at least three rounds.* ***Axis constraint****: If Axis 7's value is "Malicious" and Axis 11's value is at or to the right of "complete fairness", then Axis 18's value must be to the left of "2 rounds".*

Theorem 6 ([5]). *For unconditional security against t maliciously corrupted players, $n/3 \leq t < n/2$, a broadcast channel is required.* ***Axis constraint****: If Axis 10's value is to the right of "Computational" and Axis 7's value is "Malicious" and Axis 9's value is to the right of $n/3$, then Axis 2's value must be "Broadcast channel".*

Theorem 7 ([18]). *For (even cryptographic) security against $\geq n/3$ maliciously corrupted players, either a trusted key setup or a broadcast channel is required.* ***Axis constraint****: If axis 7's value is "Malicious" and Axis 9's value is to the right of $n/3$, then either Axis 2's value must be "Broadcast channel," or else Axis 1's value is to the left of "No trusted setup."*

Theorem 8 ([5]). *There can be no unconditionally secure protocol against an adversary controlling a majority of parties.* ***Axis constraint****: Axis 10's value can be to the right of "Computational" only if Axis 9's value is at or to the left of $n/2$.*

Theorem 9 ([9]). *There is no protocol with UC security against a dishonest majority without setup assumptions.* ***Axis constraint****: If Axis 9's value is to the right of $n/2$ and Axis 12's value is "Universally composable," then axis 1's value must be to the left of "No trusted setup".*

Theorem 10 ([4]). *In an asynchronous environment, there is no protocol with guaranteed output secure against a fail-stop adversary corrupting $n/3$ or more parties.* ***Axis constraint****: If Axis 4's value is "Asynchronous," Axis 7's value is at or to the right of "Fail-stop," and Axis 11's value is at "Guaranteed output," then Axis 9's value must be at or to the left of $n/3$.*

Theorem 11 ([4]). *In an asynchronous environment, there is no protocol with guaranteed output secure against a malicious adversary corrupting $n/4$ or more parties.* ***Axis constraint****: If Axis 4's value is "Asynchronous," Axis 7's value is "Malicious," and Axis 11's value is at "Guaranteed output," then Axis 9's value must be at or to the left of $n/4$.*

One validation of our choice of axes is that these theorems are directly and compactly expressible in terms of them, thus giving a unified representation of the central body of knowledge of MPC. The axis constraints can easily be represented in a programming or knowledge representation language, as Section 4 shows.

3 An Extensible Protocol Database

We scored more than 60 of the most significant protocols in secure multiparty computation on our axes, integrating information from our annotated bibliography, resulting in an extensible *MPC protocol database*.

Many papers in the area include multiple protocols. We give each protocol a separate entry in the database, which is labeled by adding a suffix to the usual "alpha"-style reference. For instance, "[GMW87]-mal" refers to the protocol of [18] that is secure against a malicious adversary. The database also indicates whether an implementation of the protocol is known to exist.

The work of constructing the database motivated many revisions of our set of axes and highlighted difficulties in systematizing MPC results, some of which we discuss here.

Efficiency. As mentioned in the axis descriptions, our efficiency axes are concerned primarily with asymptotic efficiency measurements. When we populated our database, we relied on evaluations in the literature, frequently from the paper actually introducing the protocol.

The model of computation in which a protocol's functionalities are expressed can have a large impact on concrete efficiency. Historically, MPC functionalities have been expressed as circuits. The original Yao model considers Boolean circuits, while most of the current state-of-the-art MPC protocols are in the arithmetic-circuit model. Implementing these requires performing field arithmetic, and, although the size of the field elements is a constant in the security parameter, the time taken to perform field operations can have a significant impact on efficiency. Although this difference in concrete efficiency is not captured by the axes, our protocol database notes the model of computation for each protocol.

Even in the asymptotic case, comparing the efficiency of MPC protocols is an extremely difficult problem because of multiple interacting aspects of efficiency present in MPC. In selecting an actual implementation, a concrete analysis and/or empirical efficiency measurements should also be consulted.

Substitutability. One factor that makes it nontrivial to enumerate a list of MPC protocols is that many protocols described in the literature make use of subprotocols for cryptographic operations in a black-box fashion, making it possible to substitute different protocols implementing that operation. This can alter not only the performance characteristics but also the computational and environmental assumptions and security and composability guarantees of the resultant protocol. In some cases, a new and improved subprotocol can trivially be used to improve an older MPC protocol, but no published work explicitly presents the improvement; in other cases a protocol explicitly allows for black-box substitution of subprotocols, in which case it is said to be stated in a *hybrid* model. In the case of the *OT-hybrid* model, in which oblivious transfer is a black box, recent work in *OT extension* has produced significant performance gains.

An extreme case of this "substitutability" factor is the IPS compiler of Ishai *et al.* [26], which is not only in the OT-hybrid model but also allows any of a wide of of honest-majority MPC protocols to be plugged in as an "outer protocol," with the resulting protocol inheriting some (but not all) of the security properties of the outer protocol.

To address this complex issue, we have limited our systematization to represent only concrete instantiations of hybrid protocols. The axes are such that

a protocol in a hybrid model must first be "instantiated" with concrete subprotocols in order to be scored. In our database, we have sought to enumerate as many such concrete protocols as possible that are based on well-known hybrid-model protocols.

Two-party secure computation. Although much research has been done specifically addressing two-party secure computation, starting from Yao's original garbled circuits idea, it can be considered as a special case of multiparty computation, in which, if security against a malicious adversary is sought, no honest majority can be assumed (Axis 9 at $n-1$). Thus, two-party protocols can at least theoretically be compared with multiparty results in this category.

In reality, however, vast improvements in efficiency have been made for the two-party case. We note that these optimizations often come at the expense of *symmetry*: The security guarantee against cheating by one of the two parties may be weaker than for the other. For instance, one of the two parties may be able to cheat with an inverse-polynomial probability, while the other may only be able to cheat with negligible probability. Asymmetry is not included in our axis definitions, and so two-party protocols are scored by the weaker of the two sets of security guarantees. Of course, the database indicates which protocols and implementations are strictly for two parties.

4 Putting the Systematization to Work

As mentioned in Section 1, the main theorems of secure multiparty computation demonstrate essential tradeoffs involved in securely computing a functionality among distrustful parties. To leverage the information that our systematization captures about these tradeoffs and gain insight into the problem space, we experimented with visually plotting the protocol axis scores of the MPC database. However, we found that the highly categorical nature of the data makes it difficult to gain insight from a static visualization. In this section, we describe an an interactive tool we have developed for interacting with the systematization and MPC protocol database, which provides a better way of coming to grips with the multi-dimensional landscape of MPC protocols.

4.1 A prototype decision-making support tool

We have developed a prototype GUI tool, *SysSC-UI*, which reads in a protocol database of axis values and enables the user to adjust a set of sliders and checkboxes corresponding to our axes of systematization. For the tool's interface, the axes are oriented vertically rather than horizontally as in this paper, so a higher position of a slider corresponds to a stronger result. A dynamically updated *results window* displays the protocols from the database that match the specified axis values. See the screenshot in Figure 1. The source code for the desktop version is available online `https://code.google.com/p/syssc-ui/`, as well as a beta web-based version `http://work.debayangupta.com/ssc/`.

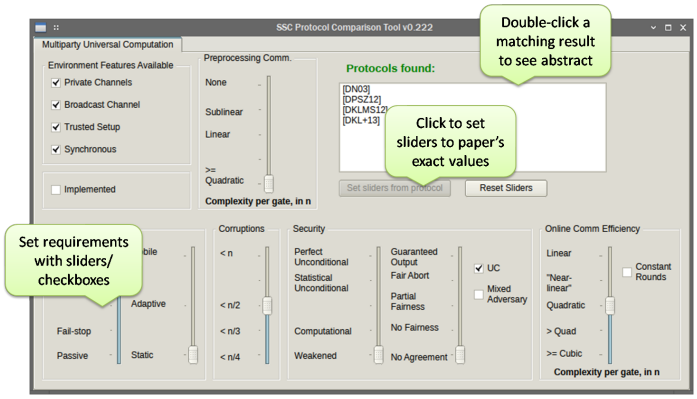


Fig. 1. Screenshot of the SysSC-UI tool for interacting with the protocol database.

For a given setting of the sliders and checkboxes, the results window shows all papers whose axis values are at the same level *or higher* than the settings. Thus, the tool presents all protocols that are *at least as good* as the specified settings. When the tool is started, the sliders and checkboxes are all set to the least constraining position, such that every protocol in the database is displayed in the results window. There is also a button to reset the tool to this state. Another button sets the sliders to the exact values of the protocol in the currently highlighted protocol, allowing the precise achievements of a protocol to be examined. This has the side effect of changing the output to the results window, so that only protocols that are at least as good as the selected one are displayed.

Double-clicking on a reference in the results window will display a pop-up window giving the authors' full names and the description of the paper containing the protocol from the annotated bibliography. The GUI also indicates when the positioning of the sliders is such that a secure computation protocol is known to be impossible, by means of an encoding of the theorems in Section 2.3.

The axes and values displayed by SysSC-UI are a subset of those in the full systematization. This was done in order to simplify the interaction and avoid confusion from the visual display of too much information at once. For example, for the composability axis, there is only a single checkbox, to indicate whether the protocol is proven universally composable or not.

We now present sample use cases highlighting the features of SysSC-UI.

4.2 Sample use cases for SysSC-UI

Finding the best protocol for a known problem. Consider a scenario in which a technology consultant is hired by a company to find a way to compute some function of a distributed set of sensitive data residing on servers owned by different divisions of the company. We show how she can use the SysSC-UI tool to find an appropriate MPC protocol for achieving this secure computation.

In the initial state of the UI tool, all four environmental assumption boxes are checked, and all sliders are in the lowest position, so that every protocol in the database is displayed in the results window. To begin, our consultant unchecks "Private Channels", knowing that the computation will be carried out over an ordinary internet connection, which should always be assumed to be tapped. She wishes to protect against adversaries that are covertly malicious, so she moves the leftmost Adversary Type slider up to "Covert". The consultant is suspicious of protocols that use a weaker model to prove security, so she moves the first Security slider up to "Computational." Furthermore, she suspects that universal composability is necessary to guarantee security in a heterogenous environment, so she checks the "UC" box. The protocols resulting from these selections can be seen in the results window in Figure 2.

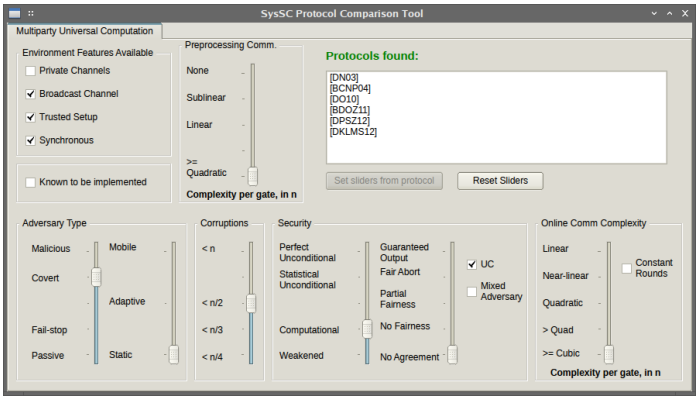


Fig. 2. Results shown by the SysSC-UI tool after selections have been made.

The consultant wishes to determine the most efficient protocol that meets these requirements, so next she moves the "Online Comm Complexity" slider up to the highest level for which the results box is not empty. We would like to achieve this online complexity with the minimum preprocessing complexity, so she tries sliding the "Preprocessing comm" slider up. If it is moved up too far, the results window becomes empty. So she readjusts both this and the Online slider to find an agreeable tradeoff between online and preprocessing complexity. The results of this exploration are shown in Figure 2.

Exposing directions for cryptographic research. In using the tool, one invariably stumbles upon a setting of the sliders / boxes that is not in the "known impossible" range, and yet has no papers with a matching protocol listed. Of course, one reason for this could simply be the incompleteness of the protocol database. Another possible reason is that the combination of features is not desirable from a security or efficiency standpoint. However, a third possibility exists, which is that a genuine opening for new research has been revealed. Two kinds of such "holes" may reveal new research directions: (1) Gaps between achieved

and proven impossible security levels, and (2) settings in which a weakening of security parameters may allow greater efficiency. An example of the second type of advance is the case of positing composable security definitions weaker than universal composability, as in [28].

5 Ongoing Work

As it is unlikely that the authors will permanently be able to stay abreast of the growing MPC literature, we have developed a web-based survey that allows researchers to submit descriptions of new protocols and their features on the axes so that they can be integrated into the protocol database and SysSC-UI. The survey is available at `http://goo.gl/T4ORzr`. Author participation is vital to the continuing usefulness of this effort.

The set of theorems in Section 2.3 should also be expanded as knowledge of secure computation increases. This work has highlighted several remaining unknowns in characterizing the possibility of secure computation; for example, perhaps some of the malicious impossibility results hold for the covert case as well. A longer-term goal is to characterize the efficiency of the protocols more precisely, in terms of the number of elementary operations, to make the efficiency of all protocols directly comparable.

Systematizations of knowledge are especially needed in research fields where a large body of results have been generated in a short time, and secure multiparty computation is undoubtedly such a field. Without an effort to systematically organize results, there may be unnecessary duplication of research efforts, the barriers to entry for new researchers may be needlessly high, and results may not see useful applications as early as they could. Our systematization of secure computation is a tool that can significantly ease the task of coming to grips with this sprawling body of results, and potentially speed its adoption in fields where it would be useful.

Acknowledgments

This material is based on research sponsored by DARPA under agreement number FA8750-13-2-0058. The U.S. government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions herein are those of the authors and should not be interpreted as necessarily representing the office policies or endorsements, either expressed or implied, of DARPA or the U.S. Government.

References

1. Aumann, Y., Lindell, Y.: Security against covert adversaries: Efficient protocols for realistic adversaries. In: Theory of Cryptography – TCC 2007. pp. 137–156 (2007)

2. Barak, B., Canetti, R., Nielsen, J.B., Pass, R.: Universally composable protocols with relaxed set-up assumptions. In: Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2004). pp. 186–195. IEEE (2004)
3. Baum, C., Damgård, I., Orlandi, C.: Publicly auditable secure multi-party computation. Cryptology ePrint Archive, Report 2014/075 (2014), http://eprint.iacr.org/
4. Ben-Or, M., Canetti, R., Goldreich, O.: Asynchronous secure computation. In: Proceedings of the 25th Annual ACM Symposium on Theory of Computing (STOC '93). pp. 52–61 (1993)
5. Ben-Or, M., Goldwasser, S., Wigderson, A.: Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract). In: Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing (STOC '88). pp. 1–10 (1988)
6. Bitansky, N., Canetti, R., Halevi, S.: Leakage-tolerant interactive protocols. In: Theory of Cryptography – TCC 2012. pp. 266–284 (2012)
7. Bogdanov, D., Kamm, L., Laur, S., Pruulmann-Vengerfeldt, P.: Secure multi-party data analysis: end user validation and practical experiments. Cryptology ePrint Archive, Report 2013/826 (2013), http://eprint.iacr.org/2013/826
8. Bogetoft, P., Christensen, D., Damgård, I., Geisler, M., Jakobsen, T., Krøigaard, M., Nielsen, J., Nielsen, J., Nielsen, K., Pagter, J., Schwartzbach, M., Toft, T.: Secure multiparty computation goes live. In: Dingledine, R., Golle, P. (eds.) Financial Cryptography and Data Security, Lecture Notes in Computer Science, vol. 5628, pp. 325–343. Springer Berlin Heidelberg (2009)
9. Canetti, R., Kushilevitz, E., Lindell, Y.: On the limitations of universally composable two-party computation without set-up assumptions. In: Advances in Cryptology – EUROCRYPT 2003, pp. 68–86. Springer (2003)
10. Cleve, R.: Limits on the security of coin flips when half the processors are faulty (extended abstract). In: STOC. pp. 364–369 (1986)
11. Cramer, R., Damgård, I.: Multiparty computation, an introduction. In: Contemporary cryptology, pp. 41–87. Springer (2005)
12. Cramer, R., Damgård, I., Nielsen, J.B.: Secure Multiparty Computation and Secret Sharing: An Information Theoretic Approach. Self-published manuscript (2013), https://users-cs.au.dk/jbn/mpc-book.pdf
13. Damgård, I., Geisler, M., Krøigaard, M., Nielsen, J.B.: Asynchronous multiparty computation: Theory and implementation. In: Public Key Cryptography. pp. 160–179 (2009)
14. Feigenbaum, J., Ishai, Y., Malkin, T., Nissim, K., Strauss, M.J., Wright, R.N.: Secure multiparty computation of approximations. ACM Transactions on Algorithms 2(3), 435–472 (2006)
15. Feigenbaum, J., Pinkas, B., Ryger, R., Saint-Jean, F.: Secure computation of surveys. In: EU Workshop on Secure Multiparty Protocols. Citeseer (2004)
16. Gennaro, R., Ishai, Y., Kushilevitz, E., Rabin, T.: On 2-round secure multiparty computation. In: Yung, M. (ed.) Advances in Cryptology – CRYPTO 2002, Lecture Notes in Computer Science, vol. 2442, pp. 178–193. Springer Berlin Heidelberg (2002)
17. Goldreich, O.: The Foundations of Cryptography - Volume 2, Basic Applications. Cambridge University Press (2004)
18. Goldreich, O., Micali, S., Wigderson, A.: How to play any mental game or a completeness theorem for protocols with honest majority. In: Proceedings of the Nine-

teenth Annual ACM Symposium on Theory of Computing (STOC '87). pp. 218–229 (1987)
19. Goldwasser, S., Kalai, Y.T., Popa, R.A., Vaikuntanathan, V., Zeldovich, N.: Reusable garbled circuits and succinct functional encryption. In: Proceedings of the 45th Annual ACM Symposium on Theory of Computing (STOC '13). pp. 555–564 (2013)
20. Goldwasser, S., Lindell, Y.: Secure computation without agreement. In: Proceedings of the 16th Int'l Symposium on DIStributed Computing (DISC). pp. 17–32 (2002)
21. Gordon, S.D., Katz, J., Kolesnikov, V., Krell, F., Malkin, T., Raykova, M., Vahlis, Y.: Secure two-party computation in sublinear (amortized) time. In: ACM Conference on Computer and Communications Security (ACM CCS 2012). pp. 513–524 (2012)
22. Goyal, V., Mohassel, P., Smith, A.: Efficient two party and multi party computation against covert adversaries. In: Advances in Cryptology – EUROCRYPT 2008. pp. 289–306 (2008)
23. Hazay, C., Lindell, Y.: Efficient Secure Two-Party Protocols – Techniques and Constructions. Information Security and Cryptography, Springer (2010)
24. Hirt, M., Lucas, C., Maurer, U., Raub, D.: Graceful degradation in multi-party computation (extended abstract). In: 5th International Conference on Information Theoretic Security (ICITS 2011). pp. 163–180 (2011)
25. Ishai, Y., Kushilevitz, E., Ostrovsky, R., Sahai, A.: Cryptography with constant computational overhead. In: Proceedings of the 40th Annual ACM Symposium on Theory of Computing (STOC '08). pp. 433–442. ACM, New York, NY, USA (2008)
26. Ishai, Y., Prabhakaran, M., Sahai, A.: Founding cryptography on oblivious transfer - efficiently. In: Advances in Cryptology – CRYPTO 2008. pp. 572–591 (2008)
27. Perry, J., Gupta, D., Feigenbaum, J., Wright, R.N.: The secure computation annotated bibliography (2014), `http://paul.rutgers.edu/~jasperry/ssc-annbib.pdf`
28. Prabhakaran, M., Sahai, A.: New notions of security: achieving universal composability without trusted setup. In: Proceedings of the 36th Annual ACM Symposium on Theory of Computing (STOC '04). pp. 242–251 (2004)
29. Rabin, T., Ben-Or, M.: Verifiable secret sharing and multiparty protocols with honest majority (extended abstract). In: Proceedings of the 21st Annual ACM Symposium on Theory of Computing (STOC '89). pp. 73–85 (1989)
30. Yao, A.C.C.: Protocols for secure computations (extended abstract). In: Proceedings of the 23rd Annual IEEE Symposium on Foundations of Computer Science (FOCS 1982). pp. 160–164 (1982)
31. Yao, A.C.C.: How to generate and exchange secrets (extended abstract). In: Proceedings of the 27th Annual IEEE Symposium on Foundations of Computer Science (FOCS 1986). pp. 162–167 (1986)