Maintaining Sublinear Locality Over Time: Adaptively Secure MPC on a Reusable Hidden Graph*

Elette Boyle[0009−0002−0360−8129]1,2†, Ran Cohen[0000−0002−1293−552X]1‡, and Pierre Meyer[0000−0003−2128−5204]3§

[1] Reichman University, Herzliya, ISRAEL. {elette.boyle,cohenran}@runi.ac.il
[2] NTT Research, Sunnyvale, USA.
[3] Aarhus University, DENMARK. pierre.meyer@cs.au.dk

Abstract. Communication *locality* of an n-party protocol measures the maximum degree of the communication graph induced by the protocol execution. While secure multi-party computation (MPC) with small, sublinear locality exists in the static-corruption setting, this goal seems nearly paradoxical in the *adaptive*-corruption setting: Even against fail-stop adversaries, small neighbour sets of honest parties lie vulnerable to identification and corruption.

Surprisingly, Chandran et al. [ITCS '15] showed that for a single MPC execution, sublinear locality and adaptive security can be simultaneously achieved, assuming honest-to-honest channels are hidden from the adversary. Their solution works in the "hidden-graph model," where a fresh, initially hidden, low-degree graph is being used in each round. In turn, the combined degree grows with every round—inherently limiting the approach to a single-shot MPC execution, and sublinear total rounds. This raises the following question, which is the focus of our work:

Is it possible to maintain sublinear locality over an unbounded number of executions *facing adaptive adversaries?*

In this work, we provide an affirmative answer in two settings:

- First, we consider semi-honest adversaries and information-theoretic security, and construct reusable MPC with $\mathsf{polylog}(n)$ locality.
- Second, we consider fail-stop adversaries and computational security, and construct reusable MPC with $\tilde{O}(n^{2/3})$ locality.

Our results are obtained by devising low-locality protocols while hiding important information about the graph topology, enabling the parties to reuse a *single hidden graph*. As an independent contribution, this serves as new results for adaptively secure topology-hiding computation (Moran, Orlov, Richelson [TCC '15]).

* Part of this work was conducted while the authors were visiting the Simons Institute for the Theory of Computing.

† Research supported in part by AFOSR Award FA9550-21-1-0046 and ERC Project HSS (852952).

‡ Research supported in part by NSF grant no. 2055568 and by ISF grant 1834/23.

§ Research supported by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme under grants agreement number 852952 (HSS), 803096 (SPEC), and 101124977 (DECRYPSIS), and by a grant from the STIBO foundation.

Table of Contents

Maintaining Sublinear Locality Over Time: Adaptively Secure MPC on a Reusable Hidden Graph . . . 1
Elette Boyle, Ran Cohen, and Pierre Meyer

1 Introduction **3**
1.1 Our Results . . . 4
1.1.1 Information-theoretic security against semi-honest adversaries. . . . 4
1.1.2 Computational security against fail-stop adversaries. . . . 5

2 Technical Overview **6**
2.1 The Distributional THB Protocol . . . 8
2.2 Correctness and Consistency . . . 9
2.3 Leakage Analysis . . . 10

3 Preliminaries **10**

4 Reusable and Adaptively Secure MPC **11**
4.1 Reusable and Adaptively Secure MPC with Low Locality . . . 11
4.2 Semi-Honest: Reusable MPC from All-To-All SMT . . . 13
4.3 Fail-Stop: Reusable MPC from Broadcast . . . 14

5 Distributional Topology-Hiding Computation **14**
5.1 Distributional Topology-Hiding Computation . . . 15
5.2 Semi-Honest Distributional TH All-To-All Communication . . . 17
5.3 Fail-Stop Distributional TH Broadcast . . . 18

6 Adaptively Secure Distributional THB **19**
6.1 Distributional THB in the Neighbourhood MPC Hybrid Model . . . 19
6.1.1 The Protocol . . . 19
6.1.2 Main Security Arguments . . . 25
6.2 Instantiating the Hybrid Model . . . 28

7 Core Lemmata **29**
7.1 Distributed Sampling of an Erdös–Rényi Graph . . . 29
7.2 Structural properties of Erdös–Rényi graphs . . . 30
7.2.1 An honest party's neighbourhood has a constant fraction of honest parties. . . . 30
7.2.2 Almost every node has a constant fraction of honest neighbours. . . . 31
7.2.3 All high-degree induced subgraphs have diameter at most three. . . . 32
7.2.4 High-degree nodes are within two hops of almost all nodes. . . . 32
7.2.5 A party with $\Omega(np)$ honest neighbours has a common neighbour with each and every honest party. . . . 33

1 Introduction

Secure multiparty computation (MPC) [Yao82, GMW87, BGW88, CCD88] enables a set of n parties to compute a joint function of their private inputs while revealing nothing but the output, even when up to t of them might collude and attack the protocol. This is a general-purpose tool for computing on sensitive data while eliminating single points of failure. The design and optimisation of MPC has been the subject of a large body of research in the last 40 years.

A natural metric, which has recently attracted attention, is the number of parties that each party must communicate with during the course of the computation. This is referred to as the *communication locality* of the protocol [BGT13]. Understanding when protocols with sublinear locality (in n) can and cannot be achieved facing linear corruptions (*i.e.* if a constant fraction of parties may be corrupted) has been an important focus of works within both distributed computing and MPC literatures.

Sublinear locality against static *corruptions.* The question was originally explored assuming communication along edges of a *fixed* incomplete graph, where corruptions can take place as a function of the graph structure; however, this setting faces impossibilities for achieving standard security [Dol82, FLM86]. These impossibilities no longer hold when each party may communicate with each other party, but chooses with whom to actually talk during the protocol's execution, *i.e.* when the communication graph evolves dynamically. This "dynamic-graph" setting opens a slew of techniques, such as randomly choosing neighbour sets and transferring information safely via small elected committees. Since corruptions are static, one can argue that high fractions of selected neighbours and elected committees must be honest.

Ultimately, building on distributed protocols such as [KSSV06, KS10, KLST11, BLZL+25], protocols secure against linearly many static corruptions have been constructed in this model with sublinear (and as low as $\mathsf{polylog}(n)$) locality, both in the context of broadcast and Byzantine Agreement [KSSV06, KS10, KLST11, BCG21, BBCL23, FGK24, GK24] and in the context of MPC [BGT13, BCP15, DKM+17], even with information-theoretic security [BCDH18].

Sublinear locality against adaptive *corruptions?* The picture drastically changes when considering adaptive corruptions, where at a first glance, the goal itself may even sound suspiciously hopeless. Extending the above positive results to the adaptive-corruption regime would seem to reintroduce the very challenges of the fixed-graph setting, where small dynamically elected sets again lie vulnerable to identification and corruption.

Indeed, the same lower bounds on locality do carry over within MPC models presuming that the adversary is privy to the set of active honest-to-honest communication edges. For example, given even fail-stop corruptions, the adversary can simply corrupt and crash all parties who receive a message from a targeted honest party. Even semi-honest adversaries can isolate an honest party and break privacy when considering information-theoretic security in the plain model (with no setup assumptions). Adaptive security faces strong lower bounds in this setting [ACD+19, BCDH18, BBCL23].

Single-shot adaptive MPC in the hidden-channels model. However, in many settings (and especially in large-scale MPC), it is overly pessimistic to assume that the adversary sees activity of pairwise communication channels between parties not under its control. Correspondingly, Chandran et al. [CCG+15] put forth a (perhaps even more natural) model, in which the adversary does *not* see which honest-to-honest communication channels are exercised, and where the aforementioned adaptive attacks no longer hold. We refer to this as the *hidden-channels model*. Surprisingly, they showed that sublinear locality (and even $\mathsf{polylog}(n)$ locality) *can* be achieved with adaptive security within this model, albeit for a single MPC execution.

Chandran et al. [CCG+15] cleverly avoided the above-discussed adaptive-corruption attacks by introducing a "hidden-graphs" technique. In this approach, the protocol freshly samples a new low-degree graph G_i for each round i of communication, such that each party only learns its neighbour-set in G_i, but otherwise G_i itself is kept hidden until used. This can be achieved given a secret-key infrastructure (SKI), in which each pair of parties receives a common secret key. Effectively, Chandran et al. first establish an all-to-all reliable message transmission (RMT) mechanism, where the parties flood messages out round by round to their (continually changing) neighbour sets. Neighbours in round i are defined by G_i, ignoring existence of prior G_j for $j < i$. This round-to-round independence loosely allows them to emulate a static-corruption setting, ensuring that future communication-edge

selection is independent of any information gleaned by the adversary as a function of the protocol up to this step.

Reusable adaptive MPC? Unfortunately, a consequence of the many-hidden-graphs approach of [CCG+15] is that the overall locality of their protocol grows with the number of *total rounds* of protocol communication. This suffices for the restricted goal of a single MPC execution with computational security, by emulating a single MPC protocol with sublinear round complexity on top of this RMT infrastructure (e.g., a constant-round MPC in the broadcast model, where each broadcast is implemented by a $\mathsf{polylog}(n)$-round protocol). However, it serves as a fundamental limit of their approach toward general protocols, where parties sequentially execute multiple MPC protocols during the course of time. After a linear number of communication rounds, the expected locality of the protocol will already be linear.

This shortcoming holds in all existing adaptively secure MPC with sublinear locality [BCDH18, CGM+24], each inspired from the many-hidden-graphs approach of [CCG+15]. Without the ability to grow the communication graph over time, one is left with a daunting task of somehow protecting a single ultimately fixed low-degree graph. This leads to the following central question:

Is it possible to maintain sublinear locality over an unbounded number of sequential executions *facing linear adaptive adversaries?*

We emphasise that this remains the case, with no existing approaches to maintain sublinear locality facing adaptive corruptions, even against fail-stop adversaries, and even under cryptographic assumptions. This setting already embodies the core challenges of achieving low locality due to the adversarial ability of honest-party isolation.

In the information-theoretic case (without cryptographic assumptions and without trusted setup), the situation is even worse: sublinear locality is not even known for a *single-shot* MPC with *semi-honest* security. Indeed, for high-depth computations the only known MPC protocols require high round complexity, invalidating the many-hidden-graphs approach altogether.

1.1 Our Results

We present a positive answer to the above question, in two primary settings:

1. Information-theoretic security against *semi-honest* corruptions, and
2. Computational security against *fail-stop* crashes.

Our results are obtained by devising low-locality protocols while hiding important information about the graph topology, enabling the parties to reuse a single hidden graph. Our main technical tools are based on "topology-hiding computation" [MOR15] which is MPC that hides the underlying communication graph. As an independent contribution, we construct new adaptively secure protocol in the *distributional* THC framework of [BBC+19].

In the following, by *reusable* general secure function evaluation (SFE) protocol, we mean security that is maintained under *any* (not a priori bounded) polynomially many sequential executions, for an arbitrary sequence of functions, while maintaining the quoted locality. Further, as standard in the literature on large-scale MPC, $\widetilde{O}$ notation hides factors of $\log n$ and (when relevant) the computational security parameter λ. We recall that our results are within the hidden-channels model of [CCG+15], where the adversary does not see which honest-to-honest communication channels are used.

1.1.1 Information-theoretic security against semi-honest adversaries.

We begin with our information-theoretic result, which guarantees $\mathsf{polylog}(n)$ locality against adaptive semi-honest corruption of nearly half of all parties.

Theorem 1 (IT semi-honest). *Let $n \in \mathbb{N}$, let $0 < \varepsilon < 1/2$ be a constant, and let $t = \varepsilon n$. There exist a reusable n-party SFE protocol against adaptive semi-honest t-adversaries with information-theoretic security in the plain model and over hidden channels, maintaining $\mathsf{polylog}(n)$ locality.*

The starting point for our protocol is an observation from [BBC+19], showing that if one can establish a protocol for all-to-all secure communication (that is, every party can securely communicate with every other party) that offers information-theoretic security, has sublinear locality, and hides certain aspects of the topology of the underlying communication graph, then it is possible to achieve sequential composition and reuse this protocol within the framework of [CCG+15]. However, [BBC+19] did not present a candidate protocol for this task (nor does one exist to date). We close this gap by constructing a protocol satisfying these requirements, inspired by Chaum's DC-nets [Cha88].

At a high level, the parties in our protocol operate over a hidden Erdös–Rényi graph where each edge appears with probability $\mathsf{polylog}(n)/n$. Such a graph can be jointly sampled by the parties in the semi-honest setting *without setup* (as observed in [CGO12a, CGO12b, CCG+15], and which we recall in Lemma 20). To enable P_i to privately send a message m to P_j, each party additively secret shares a value with its neighbour-sets: P_i shares m, P_j shares a random mask value r, and all other parties share 0. Next, each party sums up the shares obtained from its neighbours; that is, the parties jointly hold secret shares of the value $m + r$. In the last step, P_j must obtain the sum of all these shares, and recover m.

If we can guarantee the adversary will never corrupt a nontrivial cut in the Erdös–Rényi graph, disconnecting the honest communication subgraph via its corruptions, then the collection of all parties' resulting shares of $m + r$ can be simulated from just the final sum. In this case, it will suffice for parties to simply use a reliable message transmission (RMT) and communicate their shares to P_j in a privacy-free manner.

However, one must take care within this RMT phase. Despite the lack of private inputs, revealing information about the *graph topology* may enable an adversary to adaptively corrupt a cut and disconnect the honest connected component. Unfortunately, there is no known protocol for achieving RMT in Erdös–Rényi graphs with the necessary form of topology hiding.

To fix this issue, we observe that in the context of low-locality MPC, nothing forces us to have the parties communicate their shares within the same graph used to generate them. We implement the semi-honest RMT by augmenting the hidden graph with a fixed, *publicly known* cycle $C = ①\text{-}②\text{-}\ldots\text{-}ⓝ\text{-}①$ between the parties (a 1-connected path would also suffice). The parties send and sum their shares along the (fixed) path of length $n - 1$ from P_{j+1} to P_j.

Ultimately, the communication graph of our protocol operates in the union of the two graphs, the Erdös–Rényi graph and the cycle C, and remains this way throughout the course of any (polynomially many) executions of SFE. The role of the Erdös–Rényi graph is to protect the privacy of secret messages; in turn, its topology must remain hidden. We prove that this hiding is preserved as long as the honest parties form a single connected component, in which case the "DC net" messages appear simply as random values independent of the graph structure. And further, as long as the topology is hidden, then with overwhelming probability the adversary will not be able to disconnect the honest component. The role of the cycle graph C, in contrast, is to enable communication of *public* values. As such, there is no issue with the graph being disconnected by (semi-honest) corruptions, and hence its identity need not be hidden. Collectively, the two parts combined translates to the required form of distributionally topology-hiding protocol for the joint graph class of the Erdös–Rényi graph and the cycle C.

1.1.2 Computational security against fail-stop adversaries. Note that the information-theoretic "topology-hiding" protocol above completely breaks if the adversary can crash parties. Although the Erdös–Rényi graph is kept hidden, the cycle C (that is used as a privacy-free RMT channel) is publicly known, so the adversary can disconnect it by crashing any one of the parties. This demonstrates the challenge of combining adaptive security and sublinear locality: where if the adversary obtains enough leakage on the underlying (sub)graph, and identifies a sublinear cut, it can crash all parties on the cut and disconnect the graph. This calls for a drastically different approach from the semi-honest case.

Our second result achieves reusable MPC facing adaptive fail-stop adversaries. We do so by constructing a similar flavour of "topology-hiding" broadcast (THB) in the spirit of [BBC+19]), albeit in the computational setting and with partial information *leakage* about the underlying graph. Fortunately, we show that this leakage is sufficiently benign, and still enables sequential executions. This ends up requiring a delicate balance and analysis, ensuring at each step that: (1) If the graph still retains certain structural properties given the current state of corruptions and crashes, then the amount of leakage on the graph structure can be appropriately bounded; (2) Given this nontrivial amount

of leakage, then with high probability the adversary will still be unable to violate the corresponding graph property. We defer a deeper discussion to the Technical Overview.

Theorem 2 (Computational fail-stop). *Let $n \in \mathbb{N}$, let $0 < \varepsilon < 1/2$ be a constant, and let $t = \varepsilon n$. Assume the existence of adaptively secure (semi-honest) oblivious transfer. There exists a reusable n-party SFE protocol against PPT adaptive fail-stop t-adversaries with computational security in the plain model and over hidden channels, maintaining $\widetilde{O}(n^{2/3})$ locality.*

It is instructive to compare our results with the lower bounds of Chandran et al. [CGM+24]. They showed that secure data erasures are essential for a certain class of low-locality communication protocols; namely, "store-and-forward" protocols, which are a type flooding protocols in which the message identifies the sender (as used by [CCG+15]). These protocols are vulnerable to a fail-stop adversary that can passively corrupt a random subset of the parties, and after learning the message trace back and crash parties all the way back to the sender. For this reason, [CCG+15] required that when a party P_i starts sending messages to the network, the adversary cannot corrupt P_i until all recipients obtain the messages and until P_i erased the identity of the recipients from its state (*i.e.* atomic-multisend and data erasures in one atomic operation). Chandran et al. [CGM+24] used heavy machinery, combining non-committing encryption (NCE) and fully homomorphic encryption (FHE) in a clever way, to hide the sender's identity from the message and avoid this attack without assuming atomic-multisend and data erasures. Our protocols also do not assume atomic-multisend and data erasures, by using a different approach, based on topology-hiding computation. See Table 1 for a summary of our results and a comparison to prior work.

	adversary	**locality**	**max rounds**	**assumptions**	**setup**	**erasures**	**atomic multisend**
[CCG+15]	malicious	$\tilde{O}(n^{1/2})$	$o(n)$	NCE	PKI	✓	✓
[CCG+15]	malicious	polylog(n)	$o(n)$	NCE	PKI + SKI	✓	✓
[BCDH18]	malicious	polylog(n)	$o(n)$	NCE	PKI + SKI	✓	✓
[CGM+24]	malicious	polylog(n)	$o(n)$	NCE	PKI + SKI	✓	✗
[CGM+24]	malicious	polylog(n)	$o(n)$	NCE + FHE	PKI + SKI	✗	✗
this work	semi-honest	polylog(n)	**any poly(n)**	-	-	✗	✗
	fail-stop	$\tilde{O}(n^{2/3})$	**any poly(n)**	adaptive OT	-	✗	✗

Table 1: Summary of our results and comparison to prior work.

Organization of the paper. In Section 2 we present a technical overview of our fail-stop protocol. Preliminaries can be found in Section 3. Reusable MPC is defined in Section 4 and reduced to the relevant communication functionalities. Distributional THC is discussed in Section 5, and our main technical result, of a distributional THB gainst fail-stop adversaries is presented and analyzed in Section 6. Due to space limitation, we defer additional related work and some of the definitions and proofs to the appendix.

2 Technical Overview

We proceed to describe our techniques, focusing on the more challenging fail-stop result.

Reusable MPC. We begin by formally defining the notion of reusable and adaptively secure MPC with sublinear locality, *i.e.* a protocol that can be executed sequentially an unbounded (polynomial) number of times while ensuring security and sublinear locality. Our notion is different than the modular composition of Canetti [Can00], which guarantees that if a protocol satisfies its security definition, then it can be executed sequentially for an unbounded number of times. As opposed to [Can00] that ensures "local" properties (security of each protocol on its own when executed sequentially), we require a "global" property, sublinear locality, which must be maintained across all instances. Clearly,

requiring each execution on its own to have sublinear locality only suffices to support a sublinear number of executions.

Toward this goal, in Section 4 we follow and extend the ideas from [BBC+19]. We define an ideal reactive SFE functionality, such that given any q polynomial in λ and any vector of n-party functions $\mathbf{f} = (f^1, \ldots, f^q)$, the functionality $\mathcal{F}_{\mathsf{sfe}}^{\mathbf{f}}$ operates in q phases, where the jth phase acts like a (standard) SFE of f^j. We allow adaptive corruptions at any point during the ideal computation. In the real world, we consider a generic SFE protocol π that is executed q times sequentially, where the jth execution is for computing f^j; we denote this protocol by $\pi^{\mathbf{f}}$. Finally, we require that for every q that is polynomial in λ and any vector of n-party functions $\mathbf{f} = (f^1, \ldots, f^q)$, the protocol $\pi^{\mathbf{f}}$ securely realises $\mathcal{F}_{\mathsf{sfe}}^{\mathbf{f}}$.

Topology-hiding computation (THC). Our main technical tool for realising reusable and adaptively secure MPC with sublinear locality is topology-hiding computation (THC) [MOR15]. These are MPC protocols that operate over an incomplete communication graph, where, in addition to providing input privacy, the topology of the graph is also kept hidden. Specifically, every party knows its own neighbour set, and the view of the adversary can be simulated given the corrupted parties' inputs, outputs, and neighbour sets. A long line of work on THC provided strong feasibility results in the semi-honest setting, *e.g.* [MOR15,HMTZ16,AM17,ALM17,BBC+19,BBC+20,BBKM23,BBC24].

Ball et al. [BBC+19] observed that the standard definition of THC [MOR15] is too strong to be used within adaptively secure MPC in the dynamic-graph setting (as long as the graph is sufficiently sparse, *e.g.* contains a sublinear cut), as this security notion aims to protect against *worst-case* graphs. In this setting, the environment knows the graphs and can signal the adversary which parties to corrupt in order to disconnect the graph. For this reason, [BBC+19] put forth the weaker notion of *distributional THC*, where the environment is only aware of the distribution from which the graph is sampled, but not of the actual graph. This is the notion of THC we utilise in this work.

Reusable MPC from distributional TH Broadcast. The first step toward proving Theorem 2, is reducing the problem of reusable SFE to that of distributional THB. We do so in three steps:

- (*Reusable BC* $\Rightarrow$ *Reusable SFE*). First, we show that reusable SFE can be realised given reusable broadcast. That is, in a hybrid model where a reactive ideal functionality supports an unbounded number of broadcast operations. This reduction is computationally secure and assumes the existence of adaptively secure key agreement, which is used to generate (over broadcast) a sufficiently long key for one-time pad (OTP) between every pair of parties. Next, given a (standard) honest-majority adaptively secure SFE, *e.g.* [CDD+99,DI05], the parties can send their sensitive messages over broadcast by encrypting them with OTP, which is a symmetric variant of non-committing encryption.
- (*Reusable Distrib-THB* $\Rightarrow$ *Reusable BC*). Second, we show that reusable broadcast can be realised with locality $\tilde{O}(n^{2/3})$ given distributional, topology-hiding version of reusable broadcast, where the graph distribution samples an Erdös–Rényi graph with parameter $p = n^{-1/3} \cdot \log^2 n$ and where the adversary can ask for leakage about the graph. The specific leakage function Leak is as follows: the simulator may query up to $p^{-2} \log^2 n$ pairs of sets $(S_1, S_2) \in 2^{[n]} \times 2^{[n]}$, and get in return the number of edges in the initial graph (before all crashes) from the set $S_1 \times S_2$. We note that this reduction translates between two models: the first is the dynamic-graph model where each party can communicate with each other party and the graph evolves over time, and the second is the distributional THC model in which the graph is fixed but hidden.
  Initially, the parties execute the hidden-graph sampling from [CGO12a, CGO12b, CCG+15] to agree on their neighbour sets, and later they run the distributional THB protocol while communicating with their neighbour sets and ignoring all other edges.
- (*1-time Distrib-THB* $\Rightarrow$ *Reusable Distrib-THB*). Third, we show that a distributional, topology-hiding version of reusable broadcast, can be realised with given distributional, topology-hiding version of single-shot broadcast, with the same graph distribution and same leakage. This is based on a hybrid argument in which we additionally show that the specific leakage we consider does not accumulate too much over polynomially many sequential executions.

The remaining task is to construct a distributional THB protocol (for one instance of broadcast) against fail-stop adversaries for the Erdös–Rényi distribution with parameter $p = n^{-1/3} \log^2 n$ and the leakage Leak (described above). This is what we next describe.

2.1 The Distributional THB Protocol

As an initial starting point, consider the trivial protocol of *flooding broadcast*: in the first round, the broadcaster sends the message to its neighbours; then, in successive rounds, each party forwards the message to their neighbours when they receive it for the first time. While this protocol can indeed be run in an "unknown topology" setting (*i.e.* it does not require the parties to know the graph beyond their neighbourhoods), it is however very much *not* topology-hiding. Specifically, each party learns:

- *Distance:* their own distance d to the broadcaster (they receive the message in round d).
- *Direction:* which of their neighbours are at distance $d-1$ from the broadcaster (these are the parties from whom they will receive the message).

Such leakage about the topology can be devastating, as the adversary could *e.g.* corrupt a certain number of parties uniformly at random, learn the identities of neighbours which are one step closer to the broadcaster, corrupt them, and iterate until they reach neighbours of the broadcaster. Indeed, this is precisely the intuition of the lower bound from [CGM+24]. Given sufficiently many corruptions, all neighbours of the honest broadcaster will be uncovered this way, and their crashing would disconnect the broadcaster from the network.

"Secure flooding." A variation on the above flooding protocol can limit this leakage to each party learning just their *own* distance from the broadcaster. Instead of forwarding the message directly, in every round each party performs an MPC with their neighbours which reveals the message to the center party if and only if at least one neighbour knows it. This process can be performed without revealing any information about the central party's degree or neighbour set to the participating parties (see, e.g., [MOR15] and Section 6.2). After each iteration of this process, the message reaches parties one hop further from the broadcaster, while hiding the direction from which it was received.

This raises a promising idea: if each party can already infer their exact distance to the broadcaster from their local view of the network alone, then this simple protocol of "flooding enhanced with local MPC" would already be distributionally topology-hiding. For instance, in an $\mathsf{ER}(n,p)$ graph with $p = \omega(\frac{\log n}{\sqrt{n}})$, the diameter is at most 2 with all but negligible probability, so each party initially knows its own distance from every other party.

Unfortunately, leaking each party's distance from the broadcaster is still too much leakage. The source of failure is that adaptive crashes can modify the distance between parties in ways that depend on unknown portions of the graph. Indeed, if the adversary passively corrupts a node Ⓤ, thereby learning its neighbourhood, then corrupts and crashes all but one of its neighbours Ⓥ (which remains honest) (see Figure 1 for a visual summary of this attack): an honest broadcaster is at distance 2 from Ⓤ if and only if it is a neighbour of Ⓥ. If the adversary were allowed to learn u's distance from each of the honest parties, taking turns acting as broadcasters, the adversary would learn the entire neighbourhood of the honest party v.

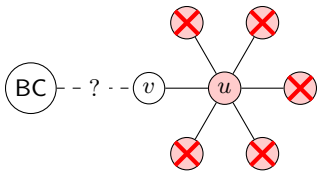


Fig. 1: "Low degree attack": the distance of corrupt u to every possible honest broadcaster BC reveals exactly honest v's neighbourhood.

Excluding low-degree parties. Fortunately, there is a way to immunise a protocol against this type of attack. The main idea is that, provided the adversary is not given too much information beyond the corrupted parties' neighbourhoods (specifically, only $o(np)$ bits of leakage on the hidden Erdös–Rényi graph), the adversary cannot identify and thus cannot corrupt more than a constant fraction of an honest party's neighbourhood.

(Informal) Core Lemma 1 (Honest parties have many honest neighbours)**.** *Let $\varepsilon, \delta > 0$. An adversary learning an (n,p)-Erdös–Rényi graph by making up to $t < \varepsilon n$ neighbourhood queries cannot*

identify $(1+\delta)\cdot tp$ *neighbours of an unqueried node with probability better than* $2^{-\mathcal{O}(tp)}$*. This probability remains negligible (in* n*) if the adversary is additionally allowed to make* $o(np)$ *arbitrary single-bit queries.*

This implies that any party with too many crashed neighbours must (with overwhelming probability) be corrupted, and we are under no obligation to ensure they receive the broadcast message. If we instruct each party to regularly ping their neighbours for a heartbeat, and simply cease any and all communication (which we will refer to as a "self-crash") if too few respond, a party will only learn its distance from the broadcaster if it still has $\Omega(np)$ live neighbours.

The protocol. The candidate protocol is now to perform "secure flooding" as described above for a given number of iterations (the analysis will show that $t+3$ rounds are sufficient to achieve consistency amongst the honest parties), but with the restriction that at every round each party pings all of its neighbours, then "crashes itself" if it received fewer than δnp such pings itself, where $\delta := \frac{1+\varepsilon}{2}$ (where $\varepsilon > 0$ is the corruption threshold fraction). As it turns out, this relatively simple protocol works, and can be simulated given roughly $1/p^2$ bits of leakage about the graph (ignoring polylogarithmic factors for now). If we choose p such that $1/p^2 = o(np)$ (which translates to locality $\widetilde{\mathcal{O}}(n^{2/3})$), then this leakage will indeed not allow the adversary to corrupt too many neighbours of an honest party.

While the protocol itself is seemingly simple in retrospect, its analysis requires significant care.

2.2 Correctness and Consistency

In order to analyse the protocol's correctness (and determine how many rounds of flooding are required), it is convenient to consider the "flooding trace" of an execution. This is the directed graph, defined after the protocol has terminated, with an edge from u to v if and only if party u ever sent the broadcast message to v (even if u subsequently crashed). Note that the receivers who ever received the message are exactly those who can be reached from the sender—and furthermore, the round in which they received the message is their distance from the sender in the graph (where by "round" here we refer to hops of message communication via the local neighbourhood MPCs). Note in the fail-stop model, parties will never forward an incorrect message: Thus, consistency holds precisely if this set of receiving parties includes either every honest party or none of them.

Assume at least one honest party holds the message by the end of the protocol (either, in the case of the broadcaster, because it held it initially, or, for the other parties, because it was delivered to them in the course of the protocol), and let k be the earliest round in which this happened. There must therefore be an honest party u such that the shortest path from the sender to u exists and is of length k. By minimality of k, all nodes on this path, including the broadcaster but excluding u (unless u is also the broadcaster), must be corrupted. Therefore $k - 1 \leq t$. This implies that if the message was ever delivered to an honest party,[4] it was within the first $t + 1$ rounds.

Once this first honest party got the message, it sent the message to all its honest neighbours,[5] who then sent it to all their honest neighbours. As it turns out, provided the adversary is only ever given $o(np)$ bits of leakage about the graph, it will (with overwhelming probability) not succeed in increasing the distance between any pair of honest parties beyond two. Skipping ahead, however, the security proof will require a slightly stronger version of this statement: any party with $\Omega(np)$ honest neighbours (which with overwhelming probability includes any honest party, by (Informal) Core Lemma 1) must, despite the adversarially caused crashes, be at distance at most two from all honest parties.

(Informal) Core Lemma 2 (Parties with $\Omega(np)$ honest neighbours are within two hops of every honest party)**.** *Let* $\varepsilon, \delta > 0$*. Consider an adversary learning an* (n, p)*-Erdös–Rényi graph by making up to* $t < \varepsilon n$ *neighbourhood queries. Let* u *be an unqueried node, and let* v *be a node whose neighbourhood contains at* $\Omega(np)$ *unqueried nodes (*v *itself may or may not have been queried). The probability that all the common neighbours between* u *and* v *have been queried is at most* $2^{-\mathcal{O}(tp)}$*. This probability remains negligible (in* n*) if the adversary is additionally allowed to make* $o(np)$ *arbitrary single-bit queries.*

[4] More precisely: to a party who will not have been corrupted by the end of the protocol.

[5] Recall that with overwhelming probability honest parties cannot "self-crash" due to having low degree, so an honest party will indeed send the message to its neighbours.

2.3 Leakage Analysis

All that remains is showing that our protocol can indeed be simulated with $\widetilde{\mathcal{O}}(1/p^2)$ bits of leakage, even after polynomially many instances of the broadcast protocol, possibly with different broadcasters.

Correctly simulating the protocol boils down to determining each corrupted receiver's output from each of the "neighbourhood MPCs". Note that if a party has fewer than $\delta \cdot np$ live neighbours (we will call such parties "low-degree"), they do not get such an output. However, simulating the views of the other corrupted parties (which we will call "high-degree") requires determining dynamically—as the graph changes with adaptive crashes—which parties are supposed to receive the message in which round.

Our simulation strategy is to juggle between two facts. On the one hand, we show that while the leakage remains bounded by $o(np)$, a certain number of invariants about the graph (*e.g.* that any pair of honest parties will be at distance at most 2) hold with all but negligible probability, and that these invariants are enough to simulate the distance between *almost every pair* of nodes. On the other hand, we show that the pairs of parties whose distance *cannot* be simulated in this way form a specific kind of sparse subgraph. This subgraph is so sparse in fact, that we can show that (n, p) Erdös–Rényi graphs do not admit such structures of size larger than $O(p^{-2}\log^2 n)$. This allows us to argue that (regardless of how the adversary chooses which nodes to corrupt), the simulator will never need more than $\widetilde{O}(p^{-2}) \ll np$ bits of leakage to simulate the protocol, regardless of how many rounds it runs for.

We refer to Section 6 for the technical details of our protocol and its analysis.

3 Preliminaries

Notations. For $n \in \mathbb{N}$, let $[n] = \{1, \cdots, n\}$. We denote by λ the security parameter. Let poly denote the set of all positive polynomials and let PPT denote a probabilistic algorithm that runs in *strictly* polynomial time. A function $\nu\colon \mathbb{N} \to \mathbb{R}$ is *negligible* if $\nu(\lambda) < 1/p(\lambda)$ for every $p \in \mathsf{poly}$ and sufficiently large λ. Given a random variable X, we write $x \leftarrow X$ to indicate that x is selected according to X, and given a non-empty, finite set $\mathcal{S}$ we write $s \leftarrow \mathcal{S}$ to indicate that s is selected uniformly at random from $\mathcal{S}$. The statistical distance between two random variables X and Y over a finite set $\mathcal{U}$, denoted $\mathrm{SD}(X, Y)$, is defined as $\frac{1}{2} \cdot \sum_{u \in \mathcal{U}} |\Pr[X = u] - \Pr[Y = u]|$.

Two distribution ensembles $X = \{X(a, \lambda)\}_{a \in \{0,1\}^*, \lambda \in \mathbb{N}}$ and $Y = \{Y(a, \lambda)\}_{a \in \{0,1\}^*, \lambda \in \mathbb{N}}$ are $\mathsf{computationally\ indistinguishable}$ (denoted $X \overset{\mathrm{c}}{\approx} Y$) if for every non-uniform polynomial-time distinguisher Adv there exists a function $\nu(\lambda) = \mathrm{negl}(\lambda)$, such that for every $a \in \{0,1\}^*$ and every $\lambda \in \mathbb{N}$,

$$\left|\Pr\left[\mathsf{Adv}(X(a,\lambda), 1^\lambda) = 1\right] - \Pr\left[\mathsf{Adv}(Y(a,\lambda), 1^\lambda) = 1\right]\right| \leq \nu(\lambda).$$

The distribution ensembles X and Y are $\mathsf{statistically\ close}$ (denoted $X \overset{\mathrm{stat}}{\approx} Y$) if for every $a \in \{0,1\}^*$ and every λ it holds that $\mathrm{SD}(X(a,\lambda), Y(a,\lambda)) \leq \nu(\lambda)$.

Graph-theoretic notations. Let $G = (V, E)$ be an undirected graph of size n, *i.e.* $|V| = n$. The *(open) neighbourhood* of a vertex v in an undirected graph G, denoted $\mathcal{N}_G(v)$, is the set of vertices sharing an edge with v in G. The *closed neighbourhood* of v in G is in turn defined by $\mathcal{N}_G[v] := \mathcal{N}_G(v) \cup \{v\}$.

MPC Model. We consider multiparty protocols in the synchronous model; we require security with guaranteed output delivery and that security remains under sequential composition. We elaborate on the model in Section 4.1, and refer the reader to [Can00, Gol04] for a precise definition of the model. Throughout the paper we assume semi-honest or fail-stop adversaries, that do not deviate from the protocol specification, but might crash. We will consider *adaptive* corruptions, where the adversary can dynamically corrupt parties during the protocol execution, learn their internal state, and potentially crash them (in case of a fail-stop adversary). In addition, we will consider both PPT adversaries (in the fail-stop setting) and computationally unbounded adversaries (in the semi-honest setting).

Recall that in the synchronous model protocols proceed in rounds, where every round consists of a *send phase* followed by a *receive phase*. The adversary is assumed to be *rushing*, meaning that it can

obtain the messages sent by the honest parties *before* sending the messages for corrupted parties; this enables the adversary to corrupt additional parties learn their state, and potentially crash them, before sending the corrupted parties' messages. We assume a complete network of point-to-point channels (broadcast is not assumed), where every party has the ability to send a message to every other party. Similarly to [CCG+15, BCDH18, CGM+24], we consider *hidden* channels, where the adversary does not get to see messages exchanged between two honest parties, and not even know whether two honest parties have communicated or not.

We do not consider *secure data erasures* nor the *atomic-multisend model* (also referred to as a weakly adaptive adversary). This means that the adversary can corrupt a party at any point during the protocol (even when it is in the middle of sending messages) and drop messages that were not delivered yet; once corrupted, the adversary learns the entire state of the party, including its input, randomness, incoming messages, and setup information (if relevant). This is the standard model for adaptive corruptions in MPC [Fel88, CFGN96, Can01] (also referred to as a strongly rushing adversary). We refer the reader to a further discussion in [HZ10, GKKZ11, CGZ23, CGM+24].

4 Reusable and Adaptively Secure MPC

In this section, we define *reusable* and adaptively secure MPC and present simple reductions to reusable communication tasks. In Section 4.1, we define *reusable* and adaptively secure MPC with sublinear locality; in Section 4.2, we consider the information-theoretic semi-honest setting, and reduce the problem to reusable all-to-all secure communication; finally, in Section 4.3, we consider the computational fail-stop setting, and reduce the problem to reusable broadcast.

4.1 Reusable and Adaptively Secure MPC with Low Locality

We start by formulating our notion of an MPC protocol that is secure under (unbounded) sequential composition. Although [Can00] already considers sequential composition with adaptive corruptions by specifying requirements on each protocol on its own, our notion is different as we require sublinear locality, which is a global property to be maintained jointly across all instances. Clearly, requiring each protocol to maintain sublinear locality independently on its own can only support sequential composition of sublinear many instances.

Toward that end, we adjust the ideal computation from [Can00] for secure function evaluation of a single function, to an ideal sequential computation of a vector of functions $\mathbf{f} = (f^1, \ldots, f^q)$, for an arbitrary q (which is polynomial in the security parameter), and for arbitrary (efficiently computable) n-party functions $f^1, \ldots, f^q$. We denote this reactive functionality by $\mathcal{F}_{\mathsf{sfe}}^{\mathbf{f}}$. We consider sequential composition by the *same* set of parties. In our definitions below, we will say that an n-party protocol π is a *generic SFE protocol* if for every efficiently computable function $f : (\{0,1\}^*)^n \to (\{0,1\}^*)^n$, the protocol π parameterised with f securely realises the "single-shot" (standard) ideal functionality $\mathcal{F}_{\mathsf{sfe}}^{f}$.

Real-world computation. Let $n, t, q \in \mathbb{N}$ be polynomial in λ, such that $t \leq n$, and let $\pi = (P_1, \ldots, P_n)$ be an n-party generic SFE protocol. For a vector of n-party functions $\mathbf{f} = (f^1, \ldots, f^q)$, we denote by $\pi^{\mathbf{f}}$ the protocol that is obtained by sequentially executing π for q instances, where the jth execution is parameterised with f^j.

More specifically, every party P_i is initialised with random coins $r_i \in \{0,1\}^*$, and operates in q phases as follows. For $j \in [q]$, the jth phase begins where every party P_i obtains its input $x_i^j \in \{0,1\}^*$ from the environment; next, the parties start interacting with one another over the specified communication network (as discussed in Section 3); finally, every honest party outputs an output value y_i^j to the environment, and the following phase begins.

The adaptive adversary Adv starts with random coins $r_{\mathsf{Adv}} \in \{0,1\}^*$ and it can corrupt a party at any point during the computation, but is restricted to corrupt at most t parties. The environment Env starts with auxiliary information z and random coins r_{Env}. Once the adversary corrupts a party, Env learns the identity of the corrupted party and hands some extra auxiliary information to Adv. Once all q phases complete, the environment may send "corrupt P_i" requests to the adversary in the post-execution corruption stage. The interaction continues until Env halts with an arbitrary output.

Definition 3 (reusable computation: real computation). *Let $n, t, q \in \mathbb{N}$ be polynomial in λ, such that $t \leq n$. Let $\pi = (P_1, \ldots, P_n)$ be an n-party generic SFE protocol, let $\mathbf{f} = (f^1, \ldots, f^q)$ be a vector of n-party functions, and let $\pi^{\mathbf{f}}$ be as described above.*

Denote by $\text{REAL}_{\pi^{\mathbf{f}},\mathsf{Adv},\mathsf{Env}}(\lambda, z, \mathbf{r})$ the output of Env resulting from the protocol interaction, as described above, on security parameter λ, auxiliary input z, and randomness $\mathbf{r} = (r_{\mathsf{Env}}, r_{\mathsf{Adv}}, r_1, \ldots, r_n)$. Let $\text{REAL}_{\pi^{\mathbf{f}},\mathsf{Adv},\mathsf{Env}}(\lambda, z)$ denote the random variable describing $\text{REAL}_{\pi^{\mathbf{f}},\mathsf{Adv},\mathsf{Env}}(\lambda, z, \mathbf{r})$, where $\mathbf{r}$ is uniformly chosen.

Ideal-world computation. Let $n, t, q \in \mathbb{N}$ be polynomial in λ, such that $t \leq n$. Let $\mathbf{f} = (f^1, \ldots, f^q)$ be a vector of length q of n-party functions to compute, where $f^j : (\{0,1\}^*)^n \to (\{0,1\}^*)^n$ for every $j \in [q]$. The ideal reactive functionality for computing $\mathbf{f}$ for parties $(P_1, \ldots, P_n)$ in the presence of an adversary (a simulator) Sim and an environment Env, is denoted $\mathcal{F}_{\mathsf{sfe}}^{\mathbf{f}}$. The environment is initialised with auxiliary information z and random coins r_{Env}. The ideal computation sequentially executes q phases, where for every $j \in [q]$, the jth phase proceeds via the following steps:

***Inputs*:** Env chooses $\mathbf{x}^j = (x_1^j, \ldots, x_n^j)$ for f^j and hands each P_i its input x_i^j.

***First corruption stage*:** Next, Sim may decide to corrupt some parties one at a time. Once a party is corrupted its inputs $(x_i^1, \ldots, x_i^j)$ and previous outputs $(y_i^1, \ldots, y_i^{j-1})$ become known to Sim. In addition, Env learns the identity of the corrupted party and hands some extra auxiliary information to Sim.

***Sending inputs to trusted party*:** An honest party P_i sends its input x_i^j to the trusted party. Upon receiving x_i^j the functionality may send some leakage to the adversary; unless explicitly specified, the leakage is $|x_i^j|$.[6]

If the adversary is semi-honest it sends x_i^j to the trusted party for every corrupted party P_i; if the adversary is fail-stop it may send to the trusted party either x_i^j or $\perp$ (if P_i sent $\perp$ in a previous iteration, it must send $\perp$). Let $\hat{x}_i^j$ be the value actually sent as the input of party P_i.

***Computation stage*:** If P_i sends no input, *i.e.* if $\hat{x}_i^j = \perp$, the trusted party sets $\hat{x}_i^j$ to be some predetermined default value. Next, the trusted party samples $r_f^j \leftarrow \{0,1\}^*$, computes $(y_1^j, \ldots, y_n^j) = f^j(\hat{x}_1^j, \ldots, \hat{x}_n^j; r_f^j)$, and sends y_i^j to party P_i.

***Second corruption stage*:** After learning the output of the jth computation, Sim may corrupt additional parties, one at a time. Upon corruption, Sim learns the corrupted party's inputs $(x_i^1, \ldots, x_i^j)$ and outputs $(y_i^1, \ldots, y_i^j)$. As before, Env learns the identity of the corrupted party and hands some extra auxiliary information to Sim.

***Outputs*:** Each honest party P_i outputs its output value y_i^j, while the corrupted parties output nothing. The adversary Sim outputs an arbitrary function of its view in the computation.

Finally, once all q computations have completed, Sim and Env proceed to the *Post-execution corruption (PEC)*, where in each round Env generates some "corrupt P_i" request, and Sim generates some arbitrary answer based on its view of the computation so far. For this purpose, Sim may corrupt more parties as described in the second corruption stage. The interaction continues until Env halts with an arbitrary output.

Definition 4 (reusable computation: ideal computation). *Let $n, t, q \in \mathbb{N}$ be polynomial in λ, such that $t \leq n$, and let $\mathbf{f} = (f^1, \ldots, f^q)$ be a vector of n-party functions. Denote by $\text{IDEAL}_{\mathbf{f},\mathsf{Sim},\mathsf{Env}}(\lambda, z, \mathbf{r})$ the output of Env resulting from the above described ideal process, as described above, on security parameter λ, auxiliary input z, and randomness $\mathbf{r} = (\mathbf{r}_f, r_{\mathsf{Env}}, r_{\mathsf{Sim}})$, where $\mathbf{r}_f = (r_f^1, \ldots, r_f^q)$. Let $\text{IDEAL}_{\mathbf{f},\mathsf{Sim},\mathsf{Env}}(\lambda, z)$ denote the probability distribution of $\text{IDEAL}_{\mathbf{f},\mathsf{Sim},\mathsf{Env}}(\lambda, z, \mathbf{r})$ where $\mathbf{r}$ is uniformly chosen.*

Security definition. We can now ready to define reusable and adaptive security of protocols. We present two definitions: one for information-theoretic semi-honest adversaries and the other for computational fail-stop adversaries.

Definition 5 (semi-honest). *Let $n, t \in \mathbb{N}$ with $t < n$. A protocol π is a reusable SFE protocol against semi-honest t-adversaries with information-theoretic security if for every q that is polynomial*

[6] the leakage is useful for modelling communication-oriented functions as done later in this section.

in λ, and every vector $\mathbf{f} = (f^1, \ldots, f^q)$ of n-party functionalities, $\pi^{\mathbf{f}}$ securely realises $\mathcal{F}_{\mathsf{sfe}}^{\mathbf{f}}$ against adaptive semi-honest t-adversaries.

That is, for every real-world adaptive semi-honest t-adversary Adv *there exists an ideal-world adaptive semi-honest t-adversary* Sim *such that for every environment* Env

$$\left\{\text{REAL}_{\pi^{\mathbf{f}},\mathsf{Adv},\mathsf{Env}}(\lambda, z)\right\}_{\lambda\in\mathbb{N},z\in\{0,1\}^*} \overset{\text{stat}}{\approx} \left\{\text{IDEAL}_{\mathbf{f},\mathsf{Sim},\mathsf{Env}}(\lambda, z)\right\}_{\lambda\in\mathbb{N},z\in\{0,1\}^*}.$$

Definition 6 (fail-stop). *Let $n, t \in \mathbb{N}$ with $t < n$. A protocol π is a reusable SFE protocol against fail-stop t-adversaries with computational security if for every q that is polynomial in λ, and every vector $\mathbf{f} = (f^1, \ldots, f^q)$ of n-party functionalities, $\pi^{\mathbf{f}}$ securely realises $\mathcal{F}_{\mathsf{sfe}}^{\mathbf{f}}$ against PPT adaptive fail-stop t-adversaries.*

That is, for every real-world PPT adaptive fail-stop t-adversary Adv *there exists an ideal-world PPT adaptive fail-stop t-adversary* Sim *such that for every PPT environment* Env

$$\left\{\text{REAL}_{\pi^{\mathbf{f}},\mathsf{Adv},\mathsf{Env}}(\lambda, z)\right\}_{\lambda\in\mathbb{N},z\in\{0,1\}^*} \overset{\text{c}}{\approx} \left\{\text{IDEAL}_{\mathbf{f},\mathsf{Sim},\mathsf{Env}}(\lambda, z)\right\}_{\lambda\in\mathbb{N},z\in\{0,1\}^*}.$$

Communication locality. The communication locality [BGT13, BCDH18] of a protocol corresponds to the maximal degree of any honest party in the communication graph induced by the protocol execution. We consider both incoming communication edges and outgoing communication edges. Note that as we do not consider malicious adversaries, corrupted parties *cannot* "spam" honest parties. We present the definition from [BCDH18] adjusted to our notation, and refer the reader to [BCDH18] for further discussions.

Definition 7 (Communication locality). *An n-party protocol π has communication locality ℓ, if for every adaptive PPT fail-stop adversary (resp., adaptive semi-honest)* Adv *corrupting up to t parties, every PPT environment* Env*, and every auxiliary input z it holds that*

$$\Pr\left[\mathsf{Degree}(\pi, \mathsf{Adv}, \mathsf{Env}, z, \lambda) > \ell\right] \leq \text{negl}(\lambda),$$

where $\mathsf{Degree}(\pi, \mathsf{Adv}, \mathsf{Env}, z, \lambda)$ is the random variable of the maximum number of parties any honest party sends messages to and receives message from, defined by running the protocol π until termination with the adversary Adv*, environment* Env*, and auxiliary input z. The probability is taken over the random coins of the honest parties, and the random coins of* Adv *and* Env*.*

4.2 Semi-Honest: Reusable MPC from All-To-All SMT

Having defined reusable MPC, we proceed to present a simple reduction of reusable and adaptively secure MPC in the semi-honest setting with information-theoretic security to the task of reusable secure all-to-all communication; namely, realizing an arbitrarily polynomially many instances of *secure message transmission* (SMT) channels between every pair of parties. For q which is polynomial in λ, we denote this functionality by $\mathcal{F}_{\mathsf{psmt}}^{q}$. This reduction follows immediately by the adaptive security for single-shot computations of the seminal BGW protocol for $t < n/2$ over secure channels [BGW88] (see also [AL17, DN14, ACS22]).

Specifically, similarly to [BBC+19], we define the *parallel secure message transmission* functionality $\mathcal{F}_{\mathsf{psmt}}$ (which in turn followed [CCGZ19, CCGZ21]), where every party P_i sends as input a vector of n messages $(m_{i\to 1}, \ldots, m_{i\to n})$ and receives as output the vector of n messages $(m_{1\to i}, \ldots, m_{n\to i})$; since the adversary is *rushing*, it learns as leakage all messages sent to corrupted parties (*i.e.* $m_{j\to i}$ with a corrupted i) before choosing the corrupted parties' inputs.[7] That is, $\mathcal{F}_{\mathsf{psmt}}$ is the SFE functionality $\mathcal{F}_{\mathsf{sfe}}^{f_{\mathsf{psmt}}}$, where f_{psmt} is the parallel SMT function described above. For $q \in \mathbb{N}$ we denote by $\mathcal{F}_{\mathsf{psmt}}^{q}$ the ideal reactive functionality for computing sequentially q instances of $\mathcal{F}_{\mathsf{psmt}}$; that is, the reactive functionality $\mathcal{F}_{\mathsf{sfe}}^{\mathbf{f}_{\mathsf{psmt}}}$, where $\mathbf{f}_{\mathsf{psmt}} = (f_{\mathsf{psmt}}^1, \ldots, f_{\mathsf{psmt}}^q)$ and each f_{psmt}^j is the parallel SMT function.

[7] We emphasise that, as opposed to static corruptions, a semi-honest adversary that is rushing can decide to corrupt additional parties *based on its incoming messages* for round r, and learn their state, before deciding on the messages to be sent by corrupted parties for round r. A fail-stop adversary may decide to crash a party without sending its messages as a result of this newly obtained information.

Lemma 8. *Let $n, t, q \in \mathbb{N}$ be polynomial in λ, such that $t < n/2$. Let $\mathbf{f} = (f^1, \ldots, f^q)$ be a vector of length q of n-ary functions to compute. Let d be the maximal multiplicative depth of the functions $f^1, \ldots, f^q$. Then, for some $q' \in O(dq)$, the functionality $\mathcal{F}_{\mathsf{sfe}}^{\mathbf{f}}$ can be realised against adaptive semi-honest t-adversaries with information-theoretic security in the $\mathcal{F}_{\mathsf{psmt}}^{q'}$-hybrid model.*

Looking ahead, in Section 5.2, we will show how to instantiate $\mathcal{F}_{\mathsf{psmt}}^{q}$ for any q that is polynomial in λ with adaptive security and sublinear locality.

4.3 Fail-Stop: Reusable MPC from Broadcast

For the fail-stop setting, we present another simple reduction of reusable and adaptively secure MPC to a different communication task: instantiating a reusable broadcast channel, denoted $\mathcal{F}_{\mathsf{bc}}^{q}$ for any q that is polynomial in λ. This is accomplished by starting with a generic, adaptively secure honest-majority MPC protocol that may communicate over pairwise secure channels and/or via a broadcast channel, *e.g.* [CDD+99, DI05], and encrypting and broadcasting every message that is intendant to be sent over a secure channel. The encryption must also be adaptively secure, so we use an *adaptively secure key agreement* protocol [CFGN96], that will be run over the broadcast channel, to establish a sufficiently long random string between every pair of parties and encrypt all communication with a one-time pad. We do not require the broadcast functionality to be corruption-fair [CGZ23], meaning that we allow the adversary to first learn an honest sender input (before any honest party) and based on this information corrupt the sender and crash it, preventing the honest parties from ever receiving this message. This is ok in our setting, as every sensitive message that is being broadcasted is first encrypted.

Specifically, we define the *broadcast* functionality $\mathcal{F}_{\mathsf{bc}}$ for a sender party P_i that sends its input message to all other parties (non-sender parties have no input). Stated differently, $\mathcal{F}_{\mathsf{bc}}$ for sender P_i is the SFE functionality $\mathcal{F}_{\mathsf{sfe}}^{f_{\mathsf{bc}}^i}$, where $f_{\mathsf{bc}}^i(x_1, \ldots, x_n) = (x_i, \ldots, x_i)$ is the broadcast function sending the ith input x_i to all (all other input values are the empty string), and the leakage send to the adversary is the entire message x_i (and not just $|x_i|$). For $q \in \mathbb{N}$ we denote by $\mathcal{F}_{\mathsf{bc}}^{q}$ the ideal reactive functionality for computing sequentially q instances of $\mathcal{F}_{\mathsf{bc}}$, where the sender is set in a round robin manner; *i.e.* the reactive functionality $\mathcal{F}_{\mathsf{sfe}}^{\mathbf{f}_{\mathsf{bc}}}$, where $\mathbf{f}_{\mathsf{bc}} = (f_{\mathsf{bc}}^1, \ldots, f_{\mathsf{bc}}^q)$ and each f_{bc}^j is the broadcast function with sender $P_{j \bmod n}$.

Lemma 9. *Let $n, t, q \in \mathbb{N}$ be polynomial in λ, such that $t < n/2$, and let $\mathbf{f} = (f^1, \ldots, f^q)$ be a vector of n-ary functions. Assume that adaptively secure key agreement exists, and let R_{ka} be its round complexity. Then, $\mathcal{F}_{\mathsf{sfe}}^{\mathbf{f}}$ can be computed in the $\mathcal{F}_{\mathsf{bc}}^{q'}$-hybrid model for some $q' = O(nq + R_{\mathsf{ka}})$ facing PPT adaptive fail-stop t-adversaries.*

Consider the constant-round protocol of Damgård and Ishai [DI05]; this is a generic SFE protocol secure against PPT fail-stop (in fact, even against malicious) t-adversaries over secure channels and a broadcast channel (where all parties can broadcast in parallel), assuming one-way functions. Denote by ℓ_{max} the maximum number of bits that a party expects to receive during the protocol when parameterised by f^j for $j \in [q]$. Initially, we augment the protocol by R_{ka} broadcast rounds, where every pair of parties establishes a one-time pad key for a total of ℓ_{max} bits.[8] Next, whenever P_i is supposed to send a message for P_k over a secure channel, P_i encrypts the message using (the relevant part of) of the shared OTP key, and broadcasts the ciphertext. This yields a protocol that requires $O(q + R_{\mathsf{ka}})$ rounds over a broadcast channel, in which all parties can broadcast in parallel at the same round. We complete the proof by serializing each parallel broadcast round via n rounds of single-sender broadcast.

Looking ahead, in Section 5.3 and Section 6, we will show how to instantiate $\mathcal{F}_{\mathsf{bc}}^{q}$ for any q that is polynomial in λ with adaptive security and sublinear locality.

5 Distributional Topology-Hiding Computation

In this section, we define our main tool for establishing reusable MPC with sublinear locality: distributional topology-hiding computation. In Section 5.1, we recall the definition of distributional THC

[8] Alternatively, assuming NCE, this phase requires a single broadcast round where every party broadcasts its NCE public key.

from [BBC+19] for the semi-honest setting, and augment it to fail-stop case. In Section 5.2, we discuss the semi-honest setting: specifically, we instantiate reusable secure all-to-all communication with sublinear locality via a distributional TH-SMT protocol over an Erdös–Rényi distribution over graphs. In Section 5.3, we discuss the fail-stop setting and reduce reusable broadcast with sublinear locality to the problem of realizing distributional THB with leakage over an Erdös–Rényi distribution over with a specific leakage function.

5.1 Distributional Topology-Hiding Computation

We begin by recalling the definition of distributional THC from [BBC+19], who focused on the semi-honest setting with information-theoretic security. We extend the definition to the fail-stop setting with leakage and computational security. Recall that [BBC+19] observed that the standard definition of THC [MOR15] is too strong to be used within adaptively secure MPC in the *dynamic graph* setting (as long as the graph is sufficiently sparse, *e.g.* contains a sublinear cut), as this security notion aims to protect against *worst-case* graphs; in this setting, the environment knows the graphs and can signal the adversary which parties to corrupt in order to disconnect the graph. For this reason, [BBC+19] put forth the notion of distributional THC, where the environment is only aware of the distribution from which the graph is sampled, but not of the actual graph.

In the setting of "classical" THC [MOR15], the communication graph is fixed but hidden from the protocol. This is captured by an ideal communication functionality, where a special "graph party" forwards the actual graph from the environment to the functionality. The functionality provides each party with its neighbour set, and is in charge of the message delivery (*i.e.* when receiving a message from some P_i to a party P_j, the functionality will forward the message if the parties are neighbours). In contrast, in the setting of distributional THC, the communication functionality samples the graph on its own. To better support compositional capabilities of this notion, [BBC+19] allowed the environment to obtain the communication graph before making its decision real/ideal; similarly to classical THC, the communication with the environment is done via the special "graph party" P_{graph}. Once the environment asks for the graph, the communication functionality enters an "out of order" state and stops processing other messages. Ball et al. [BBC+19] focused on the semi-honest setting with information-theoretic security (in the plain model), and proved that distributional THC is a non-trivial notion that is indeed weaker than classical THC.

We proceed to extend the definition of the *distributional graph communication* functionality (formally specified in Figure 2) to capture fail-stop adversaries and leakage. We remind the reader that $\mathcal{N}_G(i)$ denotes the open neighbourhood of vertex i in a graph G.

Functionality $\mathcal{F}^{\mathcal{D}}_{\mathsf{dist\text{-}graph}}$

The n-party functionality $\mathcal{F}^{\mathcal{D}}_{\mathsf{dist\text{-}graph}}$ is parameterised by a distribution $\mathcal{D}$ over graphs with vertex set $[n]$. It interacts with parties $P_1, \ldots, P_n$ as well as a special graph party P_{graph} as follows.

Initialisation Phase:

- **Input:** $\mathcal{F}^{\mathcal{D}}_{\mathsf{dist\text{-}graph}}$ receives an initialization input from each party P_i, with $i \in [n]$. Upon receiving the first such input, $\mathcal{F}^{\mathcal{D}}_{\mathsf{dist\text{-}graph}}$ samples a graph $G = (V, E) \leftarrow \mathcal{D}$, with $V = [n]$.
- **Output:** $\mathcal{F}^{\mathcal{D}}_{\mathsf{dist\text{-}graph}}$ outputs $\mathcal{N}_G(i)$ to each party P_i.

Communication Phase:

- **Input:** Whenever $\mathcal{F}^{\mathcal{D}}_{\mathsf{dist\text{-}graph}}$ receives from a party P_i a destination/data pair $(j, m_{i\to j})$, $\mathcal{F}^{\mathcal{D}}_{\mathsf{dist\text{-}graph}}$ verifies that (1) P_i did not crash, and (2) $j \in \mathcal{N}_G(i)$, proceeding to **Output** if verification passes, and ignoring this message if not.
- **Output:** If verifications passes, $\mathcal{F}^{\mathcal{D}}_{\mathsf{dist\text{-}graph}}$ sends $(i, m_{i\to j})$ to P_j.

Termination Phase:

- **Input:** $\mathcal{F}^{\mathcal{D}}_{\mathsf{dist\text{-}graph}}$ receives a termination input from P_{graph}.
- **Output:** $\mathcal{F}^{\mathcal{D}}_{\mathsf{dist\text{-}graph}}$ outputs the graph G to P_{graph} and stops processing further messages.

Crash Requests: Upon receiving a crash request for a party P_i, with $i \in [n]$, $\mathcal{F}^{\mathcal{D}}_{\mathsf{dist\text{-}graph}}$ stops processing further messages from P_i.

Fig. 2: The distributional-graph-communication functionality.

The ideal-model computation of a functionality $\mathcal{F}$ needs to be augmented to provide the simulator with the appropriate information on the graph, *e.g.* the neighbor-set of each corrupted party. Toward this purpose, [BBC+19] defined a *graph-information wrapper* functionality around $\mathcal{F}$, denoted $\mathcal{W}^{\mathcal{D}}_{\mathsf{dist\text{-}graph\text{-}info}}(\mathcal{F})$. Initially, the wrapper samples a graph from the distribution and provides every corrupted party with the neighbor-set; importantly, honest parties do not obtain their neighbor-sets unless adaptively corrupted. The reason is that otherwise, the environment (that immediately learns every value an honest party receives from the ideal functionality) will learn all neighbor-sets, and so will learn the entire graph. All other input messages are forwarded to $\mathcal{F}$ and all messages from $\mathcal{F}$ are delivered to their intendant recipients.

We augment the wrapper functionality to be parameterised with a leakage function $\mathsf{Leak}(\cdot)$ and a leakage bound B, and the adversary is permitted to ask for up to B leakage queries that are responded by applying the leakage function on the state of the functionality. If there is no leakage (*i.e.* if the function $\mathsf{Leak}(\cdot)$ always returns $\perp$), we omit its notation from the functionality.

As before, the environment can request the communication graph via a special graph party P_{graph}. After receiving this request from P_{graph}, the wrapper functionality stops processing further messages, other than corruption requests. To balance the advantage given to the environment, that now can corrupt parties as a function of the graph, after giving the graph to P_{graph}, the wrapper gives the simulator all of the input messages it received; we refer the reader to [BBC+19] for further discussion. See Figure 3 for the formal specification of this wrapper functionality.

Functionality Wrapper Functionality $\mathcal{W}^{\mathcal{D},\mathsf{Leak},B}_{\mathsf{dist\text{-}graph\text{-}info}}(\mathcal{F})$

The n-party wrapper functionality $\mathcal{W}^{\mathcal{D},\mathsf{Leak},B}_{\mathsf{dist\text{-}graph\text{-}info}}$ is parameterised by a distribution $\mathcal{D}$ over graphs with vertex set $[n]$, by a leakage function $\mathsf{Leak}(\cdot)$, and a leakage bound B. It internally runs a copy of $\mathcal{F}$ and interacts with parties $P_1, \ldots, P_n$, with a special graph party P_{graph}, and with a simulator, as follows:

Initialisation Phase:

- **Input:** $\mathcal{W}^{\mathcal{D},\mathsf{Leak},B}_{\mathsf{dist\text{-}graph\text{-}info}}(\mathcal{F})$ receives an initialisation input from each party P_i, for every $i \in [n]$. Upon receiving the first such input, $\mathcal{W}^{\mathcal{D},\mathsf{Leak},B}_{\mathsf{dist\text{-}graph\text{-}info}}(\mathcal{F})$ samples a graph $G = (V, E) \leftarrow \mathcal{D}$, with $V = [n]$.
- **Output:** $\mathcal{W}^{\mathcal{D},\mathsf{Leak},B}_{\mathsf{dist\text{-}graph\text{-}info}}(\mathcal{F})$ outputs $\mathcal{N}_G(i)$ to each corrupted party P_i.

Computation Phase:

- **Input:** $\mathcal{W}^{\mathcal{D},\mathsf{Leak},B}_{\mathsf{dist\text{-}graph\text{-}info}}(\mathcal{F})$ forwards every message it receives from the parties to $\mathcal{F}$.
- **Output:** $\mathcal{W}^{\mathcal{D},\mathsf{Leak},B}_{\mathsf{dist\text{-}graph\text{-}info}}(\mathcal{F})$ forwards every message sent by $\mathcal{F}$ to the intended recipient.

Termination Phase:

- **Input:** $\mathcal{W}^{\mathcal{D},\mathsf{Leak},B}_{\mathsf{dist\text{-}graph\text{-}info}}(\mathcal{F})$ receives a termination input from P_{graph}.
- **Output:** $\mathcal{W}^{\mathcal{D},\mathsf{Leak},B}_{\mathsf{dist\text{-}graph\text{-}info}}(\mathcal{F})$ sends a termination message to the simulator, including all input messages sent to $\mathcal{F}$, outputs the graph G to P_{graph} and stops processing further messages, except for corruption requests.

Corruption Requests: Upon receiving from the simulator a corruption request for party P_i, $\mathcal{W}^{\mathcal{D},\mathsf{Leak},B}_{\mathsf{dist\text{-}graph\text{-}info}}(\mathcal{F})$ records the corruption request and sends $\mathcal{N}_G(i)$ to P_i.

Leakage Requests: Upon receiving a leakage request from the adversary, $\mathcal{W}^{\mathcal{D},\mathsf{Leak},B}_{\mathsf{dist\text{-}graph\text{-}info}}(\mathcal{F})$ checks that fewer than B leakage requests have been answered so far (and ignores this request otherwise), then applies the function $\mathsf{Leak}(\cdot)$ to its current state (including the sampled graph and all its input messages), and responds to the adversary with the result.

Fig. 3: The distributional-graph-information wrapper functionality.

We are now ready to present the security definition of distributional THC. We present two definitions: one for semi-honest adversaries in the information-theoretic setting (without leakage), and the other for fail-stop adversaries in the computational setting (with leakage).

Definition 10 (Distributional topology hiding: semi-honest). *Let $n, t \in \mathbb{N}$ with $t < n$ and let $\mathcal{D}$ be a distribution over graphs with n nodes. A protocol π realises a functionality $\mathcal{F}$ with information-theoretic security in a distributional-topology-hiding manner with respect to $\mathcal{D}$ tolerating semi-honest t-adversaries, if π realises $\mathcal{W}^{\mathcal{D}}_{\mathsf{dist\text{-}graph\text{-}info}}(\mathcal{F})$ in the $\mathcal{F}^{\mathcal{D}}_{\mathsf{dist\text{-}graph}}$-hybrid model with information-theoretic security tolerating semi-honest t-adversaries.*

Definition 11 (Distributional topology hiding: fail-stop). *Let $n, t \in \mathbb{N}$ with $t < n$, let $\mathcal{D}$ be a distribution over graphs with n nodes, let $\mathsf{Leak}(\cdot)$ be a leakage function, and let $B \in \mathbb{N}$. A protocol π realises a functionality $\mathcal{F}$ with computational security in a distributional-topology-hiding manner with respect to $\mathcal{D}$ with leakage Leak and bound B tolerating fail-stop t-adversaries, if π realises $\mathcal{W}^{\mathcal{D},\mathsf{Leak},B}_{\mathsf{dist\text{-}graph\text{-}info}}(\mathcal{F})$ in the $\mathcal{F}^{\mathcal{D}}_{\mathsf{dist\text{-}graph}}$-hybrid model with computational security tolerating fail-stop t-adversaries.*

5.2 Semi-Honest Distributional TH All-To-All Communication

We proceed to realise the reusable parallel SMT functionality $\mathcal{F}^{q}_{\mathsf{psmt}}$ (for an arbitrary q) using a distributional TH-SMT protocol. We will use an Erdös–Rényi distribution, denoted $\mathcal{D}_{\mathsf{ER}}(n,p)$, where every edge appears in the graph with probability p. We begin by recalling the result from [BBC+19], showing that *assuming* there is a protocol for computing $\mathcal{F}_{\mathsf{psmt}}$ in a distributional-topology-hiding manner with respect to $\mathcal{D}_{\mathsf{ER}}(n,p)$ (for an appropriate p) then there is a protocol for realizing $\mathcal{F}^{q}_{\mathsf{psmt}}$ with sublinear locality. This reduction follows three steps:

1. Given a distributional-topology-hiding protocol for $\mathcal{F}_{\mathsf{psmt}}$, *i.e.* a protocol for $\mathcal{W}^{\mathcal{D}_{\mathsf{ER}}(n,p)}_{\mathsf{dist\text{-}graph\text{-}info}}(\mathcal{F}_{\mathsf{psmt}})$ in fixed and unknown communication graph with sublinear degree (modeled as $\mathcal{F}^{\mathcal{D}_{\mathsf{ER}}(n,p)}_{\mathsf{dist\text{-}graph}}$), there is a protocol for a single-instance of $\mathcal{W}^{\mathcal{D}_{\mathsf{ER}}(n,p)}_{\mathsf{dist\text{-}graph\text{-}info}}(\mathcal{F}_{\mathsf{psmt}})$ in the dynamic-graph setting (without any setup assumptions) with sublinear locality.
   This is achieved by having the parties execute the graph-sampling phase from [CCG+15] (that is based on [CGO12b]) to sample an graph from $\mathcal{D}_{\mathsf{ER}}(n,p)$ such that each party only obtains its neighbor-set. Next, the parties run the aforementioned protocol while ignoring incoming messages from other edges. Note that the graph-sampling phase from [CCG+15] requires $\tilde{O}(\sqrt{n})$ locality.
2. Let q be polynomial in λ. Given a protocol for $\mathcal{W}^{\mathcal{D}_{\mathsf{ER}}(n,p)}_{\mathsf{dist\text{-}graph\text{-}info}}(\mathcal{F}_{\mathsf{psmt}})$ in the dynamic-graph setting with sublinear locality, there is a protocol for $\mathcal{W}^{\mathcal{D}_{\mathsf{ER}}(n,p)}_{\mathsf{dist\text{-}graph\text{-}info}}(\mathcal{F}^{q}_{\mathsf{psmt}})$ in the dynamic-graph setting with sublinear locality. This is achieved by a standard hybrid argument.
3. Given a protocol for $\mathcal{W}^{\mathcal{D}_{\mathsf{ER}}(n,p)}_{\mathsf{dist\text{-}graph\text{-}info}}(\mathcal{F}^{q}_{\mathsf{psmt}})$ in the dynamic-graph setting with sublinear locality, it immediately follows that there is a protocol for $\mathcal{F}^{q}_{\mathsf{psmt}}$ in the dynamic-graph setting with sublinear locality.

We state [BBC+19, Lem. 7.1], adjusted to the notation in this paper.

Lemma 12. *Let $n, q \in \mathbb{N}$ be polynomial in λ, let $0 < \varepsilon < 1$ be a constant, and let $t = \varepsilon n$. Assume that $\mathcal{W}^{\mathcal{D}_{\mathsf{ER}}(n,p)}_{\mathsf{dist\text{-}graph\text{-}info}}(\mathcal{F}_{\mathsf{psmt}})$ can be realised with adaptive semi-honest t-adversaries and information-theoretic security in the $\mathcal{F}^{\mathcal{D}_{\mathsf{ER}}(n,p)}_{\mathsf{dist\text{-}graph}}$-hybrid model, for $p = (\log^{c}(n))/\sqrt{n}$ with some constant $c > 1$.*

Then, there exists a protocol that realises $\mathcal{W}^{\mathcal{D}_{\mathsf{ER}}(n,p)}_{\mathsf{dist\text{-}graph\text{-}info}}(\mathcal{F}^{q}_{\mathsf{psmt}})$ with adaptive semi-honest t-adversaries and information-theoretic security in the plain model with hidden communication channels using $\tilde{O}(\sqrt{n})$ locality.

Ball et al. [BBC+19] did not provide a distributional-topology-hiding protocol for $\mathcal{F}_{\mathsf{psmt}}$ though. The main contribution in this section is closing this gap. Denote by $\mathcal{D}^{\mathsf{aug}}_{\mathsf{ER}}(n,p)$ the Erdös–Rényi distribution "augmented" by a cycle $C = ①\text{-}②\text{-}\ldots\text{-}ⓝ\text{-}①$. That is, to sample from $\mathcal{D}^{\mathsf{aug}}_{\mathsf{ER}}(n,p)$, first sample a graph $G \leftarrow \mathcal{D}_{\mathsf{ER}}(n,p)$, and second, add to G every edge in the cycle C. Note that Lemma 12 holds when using $\mathcal{D}^{\mathsf{aug}}_{\mathsf{ER}}(n,p)$ instead of $\mathcal{D}_{\mathsf{ER}}(n,p)$.

Theorem 13. *Let $n, q \in \mathbb{N}$ be polynomial in λ, let $0 < \varepsilon < 1$ be a constant, and let $t = \varepsilon n$. Let $p \in \omega(\frac{\log n}{n})$. Then, $\mathcal{W}^{\mathcal{D}^{\mathsf{aug}}_{\mathsf{ER}}(n,p)}_{\text{dist-graph-info}}(\mathcal{F}_{\mathsf{psmt}})$ can be realised with adaptive semi-honest t-adversaries and information-theoretic security in the $\mathcal{F}^{\mathcal{D}^{\mathsf{aug}}_{\mathsf{ER}}(n,p)}_{\text{dist-graph}}$-hybrid model.*

Proof sketch. We start by presenting the SMT protocol.

Protocol Distributional Topology-Hiding SMT

The protocol is specified in the $\mathcal{F}^{\mathcal{D}^{\mathsf{aug}}_{\mathsf{ER}}(n,p)}_{\text{dist-graph}}$-hybrid model, and is parameterised by a sender P_S, holding an input bit x_S, and a receiver P_R.

The Protocol: Each party P_i, with $i \in [n]$, proceeds as follows:

- ***Initialisation Phase:*** Send an initialisation message to $\mathcal{F}^{\mathcal{D}^{\mathsf{aug}}_{\mathsf{ER}}(n,p)}_{\text{dist-graph}}$, and wait to receive a neighbourhood $\mathcal{N}_i \subseteq [n]$.
- ***Communication Phase:***
  - **Round $r = 1$: (DC-net, sharing across $\mathsf{ER}(n,p)$.)**
    1. The receiver P_R samples $x_R \leftarrow \{0,1\}$, every P_k with $k \notin \{S, R\}$ sets $x_k := 0$.
    2. Sample additively secret share x_i as $\{r_{i\to j}\}_{j\in\mathcal{N}_i}$.
    3. For each $j \in \mathcal{N}_i$ (in parallel) do:
       (a) Send $(j, r_{i\to j})$ to $\mathcal{F}^{\mathcal{D}^{\mathsf{aug}}_{\mathsf{ER}}(n,p)}_{\text{dist-graph}}$ // $r_{i\to j}$ is P_j's share of x_i.
       (b) Wait to receive $(j, r_{j\to i})$ from $\mathcal{F}^{\mathcal{D}^{\mathsf{aug}}_{\mathsf{ER}}(n,p)}_{\text{dist-graph}}$ // $r_{j\to i}$ is P_i's share of x_j.
    4. Set $r_i := \sum_{j\in\mathcal{N}_i} r_{j\to i}$ // r_i is P_i share of $\sum_{i=1}^{n} x_i$
  - **Round $r = 2, \ldots, n$ (DC-net: RMT along $\boxed{\mathsf{P}_{\mathsf{R}+1}}$-$\ldots$-$\boxed{\mathsf{P}_{\mathsf{R}}}$.)**
    1. Round $r = 2$: party P_{R+1} sends $(R+2, r_{R+1})$ to $\mathcal{F}^{\mathcal{D}^{\mathsf{aug}}_{\mathsf{ER}}(n,p)}_{\text{dist-graph}}$.
    2. Round $r > 2$: party P_k, upon receiving $(k-1, m_{k-1})$ from $\mathcal{F}^{\mathcal{D}^{\mathsf{aug}}_{\mathsf{ER}}(n,p)}_{\text{dist-graph}}$, sends $(k+1, m_{k-1} \oplus r_k)$ to $\mathcal{F}^{\mathcal{D}^{\mathsf{aug}}_{\mathsf{ER}}(n,p)}_{\text{dist-graph}}$.
    3. Round $r = n$: party P_R, upon receiving $(R-1, m_{R-1})$ from $\mathcal{F}^{\mathcal{D}^{\mathsf{aug}}_{\mathsf{ER}}(n,p)}_{\text{dist-graph}}$, sets $y_R := m_{R-1} \oplus r_R$.
- ***Output Phase:*** Party P_R outputs y_R.

Fig. 4: Information-theoretic adaptive distributional topology-hiding sum for the graph class $\mathsf{ER}(n,p) + P_n$, for any $p = \omega(n^{-1}\log n)$.

The proof follows from two parts: Initially, in round 1, the parties shares their values with their neighbor-sets: the sender P_S shares its input bit, the receiver P_R shares a random bit, and all other parties share zero. Since this ia an Erdös–Rényi graph with super-logarithmic degree, with overwhelming probability, none of the neighbor sets of honest parties is fully corrupted (by a simple Chernoff bound).
Second, the RMT protocol is executed by summing each party's received messages over the path of length $n-1$ $\boxed{\mathsf{P}_{\mathsf{R}+1}}$-$\ldots$-$\boxed{\mathsf{P}_{\mathsf{R}}}$. At the end of this phase the receiver P_R obtain the value $x_S \oplus x_R$ and can recover X_S, and by the perfect security of OTP, the view of every corrupted party can be simulated by a random value (with adjustments if P_S or P_R are corrupted). □

5.3 Fail-Stop Distributional TH Broadcast

The protocol in Section 5.2 completely fails facing fail-stop corruptions, as the adversary can crash the *unhidden* cycle used for broadcast. As discussed in the introduction, our approach in the fail-stop setting is different and more technically involved. We begin by defining the leakage function $\mathsf{Leak}(\cdot)$, and present two reductions: the first is realizing $\mathcal{F}^{q}_{\mathsf{bc}}$ from $\mathcal{W}^{\mathcal{D}_{\mathsf{ER}}(n,p),\mathsf{Leak},B}_{\text{dist-graph-info}}(\mathcal{F}^{q}_{\mathsf{bc}})$, and the second is realizing $\mathcal{W}^{\mathcal{D}_{\mathsf{ER}}(n,p),\mathsf{Leak},B}_{\text{dist-graph-info}}(\mathcal{F}^{q}_{\mathsf{bc}})$ from $\mathcal{W}^{\mathcal{D}_{\mathsf{ER}}(n,p),\mathsf{Leak},B}_{\text{dist-graph-info}}(\mathcal{F}_{\mathsf{bc}})$. Later, in Section 6, we will instantiate $\mathcal{W}^{\mathcal{D}_{\mathsf{ER}}(n,p),\mathsf{Leak},B}_{\text{dist-graph-info}}(\mathcal{F}_{\mathsf{bc}})$.

Lemma 14. *Let $n, q \in \mathbb{N}$ be polynomial in λ, let $0 < \varepsilon < 1$ be a constant, let $t = \varepsilon n$, and let $p = n^{-1/3}\log^2 n$. Let $B = p^{-2}\log^2 n$ and let $\mathsf{Leak}(\cdot)$ be the function which, on input a pair of sets $(S_1, S_2) \in 2^{[n]} \times 2^{[n]}$, returns the number of edges in $S_1 \times S_2$. Assume that $\mathcal{W}_{\mathsf{dist\text{-}graph\text{-}info}}^{\mathcal{D}_{\mathsf{ER}}(n,p),\mathsf{Leak},B}(\mathcal{F}_{\mathsf{bc}}^{q})$ can be realised with PPT adaptive fail-stop t-adversaries in the $\mathcal{F}_{\mathsf{dist\text{-}graph}}^{\mathcal{D}_{\mathsf{ER}}(n,p)}$-hybrid model.*
Then, there exists a protocol that realises $\mathcal{F}_{\mathsf{bc}}^{q}$ against PPT adaptive fail-stop t-adversaries in the plain model with hidden communication channels using $O(n^{2/3}\log n)$ locality.

Proof. □

Lemma 15. *Let $n, q \in \mathbb{N}$ be polynomial in λ, let $0 < \varepsilon < 1$ be a constant, let $t = \varepsilon n$, and let $p = n^{-1/3}\log^2 n$. Let $B = p^{-2}\log^2 n$ and let $\mathsf{Leak}(\cdot)$ be the function which, on input a pair of sets $(S_1, S_2) \in 2^{[n]} \times 2^{[n]}$, returns the number of edges in $S_1 \times S_2$. Assume that $\mathcal{W}_{\mathsf{dist\text{-}graph\text{-}info}}^{\mathcal{D}_{\mathsf{ER}}(n,p),\mathsf{Leak},B}(\mathcal{F}_{\mathsf{bc}})$ can be realised with PPT adaptive fail-stop t-adversaries in the $\mathcal{F}_{\mathsf{dist\text{-}graph}}^{\mathcal{D}_{\mathsf{ER}}(n,p)}$-hybrid model.*
Then, there exists a protocol that realises $\mathcal{W}_{\mathsf{dist\text{-}graph\text{-}info}}^{\mathcal{D}_{\mathsf{ER}}(n,p),\mathsf{Leak},B}(\mathcal{F}_{\mathsf{bc}}^{q})$ against PPT adaptive fail-stop t-adversaries in the plain model with hidden communication channels using $O(n^{2/3}\log n)$ locality.

Proof. □

The proof of our second main result, Theorem 2, follows from Corollary 19 and Lemmas 9, 14 and 15.

6 Adaptively Secure Distributional THB

In Section 6.1 we describe our distributional THB protocol in a hybrid model where for every party, the neighbourhood of the party can use an ideal functionality to ideally compute "conditional OR." In Section 6.2, we realise this ideal functionality over a star-shape topology.

6.1 Distributional THB in the Neighbourhood MPC Hybrid Model

6.1.1 The Protocol

Protocol Distributional topology-hiding broadcast Π_{BC}

Parties: $P_1, \ldots, P_n$

Parameters: The protocol is parameterised with

1. a number of parties n,
2. a constant $\varepsilon \in (0, 1)$ defining the corruption threshold $t \leq \varepsilon n$,
3. a sender index $i^\star \in [n]$,
4. a broadcast message space $\mathcal{M}$ and a special symbol $\perp \notin \mathcal{M}$,
5. the "conditional OR" function $\mathsf{condOR}\colon (\mathcal{M} \cup \{\perp\})^{D+1} \to \mathcal{M} \cup \{\perp\}$ is defined as follows:
   - If condOR is given fewer than $D+1$ inputs, it outputs $\perp$;
   - Otherwise, if all the inputs to condOR are $\perp$, it outputs $\perp$;
   - Otherwise, if all of the inputs in $\mathcal{M}$ are equal, it outputs this common value $m \in \mathcal{M}$;
   - Otherwise (it received at least two different inputs from $\mathcal{M}$), it outputs $\perp$.

Given the above parameters, we further define:

1. $p \leftarrow n^{-1/3}\log^2 n$ // Erdös–Rényi parameter
2. $D \leftarrow \frac{1+\varepsilon}{2}np$ // threshold for a party to be considered "high-degree"

Let $(\mathsf{sid}_{u,r})_{u\in[n],r\in[t+3]}$ be $n(t+3)$ distinct session IDs.
// identifies P_u's "neighbourhood MPC" in the rth round of flooding

Hybrid Model: $(\mathcal{F}_{\mathsf{dist\text{-}graph}}^{\mathsf{ER}(n,p)}, \mathcal{F}_{\mathsf{star\text{-}SFE}})$

Inputs: The sender $P_{i^\star}$ holds as input a message $m \in \mathcal{M}$.

The Protocol: Each party P_i (with $i \in [n]$) does the following:

– ***Initialisation Phase:***
1. Send an initialisation message to $\mathcal{F}_{\text{dist-graph}}^{\mathsf{ER}(n,p)}$, and wait to receive a neighbourhood
2. Initialise $\mathsf{live\text{-}ctr}_i \leftarrow 0$ // Counter used to track the number of...
   // ...live neighbours in each heartbeat phase
3. Initialise $\mathsf{high\text{-}degree}_i \leftarrow 1$
4. Initialise $\mathsf{out}_i \leftarrow \begin{cases} \perp & \text{if } i \neq i^\star \\ m & \text{otherwise} \end{cases}$

– ***Communication Phase:***
Rounds $1, \ldots, 2n^2(t+3)$:
// Even rounds are reserved for heartbeats
// The odd rounds are indexed by $[n] \times [n+1] \times [t+3]$:
// (1) round $(u, v, r) \in [n] \times [n] \times [t+3]$ is dedicated to P_u's ...
// ...participation in P_v's r^{th} neighbourhood MPC
// (2) round $(u, n+1, r)$ is a special round dedicated to P_u ...
// ...getting the output of its rth neighbourhood MPC

- If the round number is even: // Heartbeat
  1. Reset $\mathsf{live\text{-}ctr}_i \leftarrow 0$
  2. If $\mathsf{high\text{-}degree}_i$, send $(i, j, 0)$ to $\mathcal{F}_{\text{dist-graph}}^{\mathsf{ER}(n,p)}$ for all $j \in \mathcal{N}_G(i)$.
  3. Wait[a] to receive $(i, j, 0)$ from $\mathcal{F}_{\text{dist-graph}}^{\mathsf{ER}(n,p)}$ for each $j \in \mathcal{N}_G(i)$; increment $\mathsf{live\text{-}ctr}_i$ for each such message received.
  4. Update $\mathsf{high\text{-}degree}_i \leftarrow (\mathsf{live\text{-}ctr}_i \geq D)$

- If the round number is of the form $2((u-1) + n(v-1) + n^2(r-1)) + 1$, with $(u, v, r) \in [n] \times [n] \times [t+3]$:
  // Round dedicated to P_u's (possibly empty)...
  // ...contribution to P_v's r^{th} neighbourhood MPC
  If $(i = u) \vee (u \in \mathcal{N}_G(i))$:
  If $\mathsf{high\text{-}degree}_i$, send $(\mathsf{out}_i, \mathsf{sid}_{u,r})$ to $\mathcal{F}_{\mathsf{star\text{-}SFE}}(\mathsf{condOR})$

- If the round number is of the form $2((u-1) + n^2 r) + 1$, with $(u, r) \in [n] \times [t+3]$:
  // Round dedicated to P_u's output from its rth neighbourhood MPC
  Wait to receive a message $(\mathsf{out}, m, \mathsf{sid}_{u,r})$ from $\mathcal{F}_{\mathsf{star\text{-}SFE}}(\mathsf{condOR})$ and update $\mathsf{out}_i \leftarrow m$

– ***Output Phase:*** Output out_i .

[a] Wait until the end of this round's communication phase.

Fig. 5: Distributional topology-hiding computation over a class of Erdös–Rényi graphs.

Theorem 16 (Distributional THC in the neighbourhood-MPC hybrid model). *Let $\varepsilon \in (0, 1)$ be a constant, let $t \leq \varepsilon n$, let $\mathcal{D}$ be the distribution of Erdös–Rényi graphs of parameters (n, p) with $p = n^{-1/3} \log^2 n$, let $B = p^{-2} \log^2 n$ (the bound on the number of authorised queries), and let* Leak *be the function which, on input a pair of sets $(S_1, S_2) \in (2^{[n]})^2$, returns the number of edges in $S_1 \times S_2$.*
The protocol of Figure 5 securely realises $\mathcal{W}_{\text{dist-graph-info}}^{\mathcal{D},\mathsf{Leak},B}(\mathcal{F}_{\mathsf{BC}})$ (with statistical security) in the $(\mathcal{F}_{\text{graph}}^{\mathcal{D}}, \mathcal{F}_{\mathsf{star\text{-}SFE}}(\mathsf{condOR}))$-hybrid model in the presence of a threshold-t, adaptive, fail-sop adversary.

Proof. Let Adv be an adaptive, fail-stop adversary that interacts with parties $P_1, \ldots, P_n$ running protocol Π_{BC} (Figure 5) in the $(\mathcal{F}_{\text{graph}}^{\mathcal{D}}, \mathcal{F}_{\mathsf{star\text{-}SFE}}(\mathsf{condOR}))$-hybrid model. Consider the simulator Sim of Figure 6. Let us now show that no environment Env can distinguish with non-negligible probability whether it is interacting with Adv and Π_{BC} in the $(\mathcal{F}_{\text{graph}}^{\mathcal{D}}, \mathcal{F}_{\mathsf{star\text{-}SFE}}(\mathsf{condOR}))$-hybrid model, or with Sim and $\mathcal{W}_{\text{dist-graph-info}}^{\mathcal{D},\mathsf{Leak},B}(\mathcal{F}_{\mathsf{BC}})$.

Simulator Sim

The simulator is parameterised by an adaptive, fail-stop, p.p.t. adversary Adv. Additionally, the simulator inherits all the parameters of the protocol Π_{BC} of Figure 5 (*e.g.* n, ε, *etc*).

Simulating the communication with the environment: Every input value that Sim receives from the environment Env is written on Adv's input tape (as if coming from Adv's environment), and every output value written by Adv on its output tape is copied to Sim's own output tape (to be read by Sim's environment Env).

Simulating the protocol's execution:

1. $\mathcal{D}_{\mathsf{H}} \leftarrow \varnothing$ // Set tracking the list of honest parties
2. $\mathcal{D}_{\mathsf{SH}} \leftarrow \varnothing$ // Set tracking the list of semi-honestly corrupted parties
3. $\mathcal{D}_{\mathsf{FS}} \leftarrow \varnothing$ // Set tracking the list of crashed parties
4. $\mathcal{D}_{\mathsf{LD}} \leftarrow \varnothing$ // Set tracking the list of low-degree parties

5. Sim initialises a vector called-oracle: $[n]^2 \to \{0,1\}$ (initialised to 0)
   // Table used to store in position (i,j) if the oracle was ever called...
   // ...for the number edges between i and the honest neighbours of j

6. Sim initialises a vector oracle: $[n]^2 \to [0,n)$ (initialised to 0)
   // Table used to store in position (i,j) the number edges between i...
   // ...and the honest neighbours of j
   // This value is updated whenever a neighbour of j is corrupted...
   // ...(*i.e.* decremented if said corrupted neighbour is a neighbour of i)

7. Sim initialises a directed graph $G_{\mathsf{flooding}} = (V, E_{\mathsf{flooding}}) \leftarrow ([n], \varnothing)$; throughout the protocol (as the variable G_{flooding} changes); Sim maintains an adjacency list in: $[n] \to 2^{[n]}$ of each party's incoming edges
   // G_{flooding} is the simulator's view of the "predicted flooding trace"...
   // ...(it contains its edges which have at least one corrupted endpoint)

8. Sim activates Adv, then sends an activation message to $\mathcal{W}^{\mathcal{D},\mathsf{Leak},B}_{\mathsf{dist\text{-}graph\text{-}info}}$ and waits to receive the broadcast message $m \in \mathcal{M}$.

9. Sim then proceeds to simulate the protocol round-by-round. Upon receiving a semi-honest (resp. fail-stop) corruption request from Adv, the simulator *atomically*[a] executes subroutine "***Simulating semi-honest corruption requests***" (resp. "***Simulating fail-stop corruption requests***"). If the round is an even one, the simulator continues where it left off (*e.g.* it may have already sent a heartbeat to some of P_i's neighbours before crashing). If the round is an odd one, it involves only a predetermined party (or functionality) sending a single message to a predetermined party (or functionality): if that message was already sent before the crash was processed, the simulator resumed where it left off, but if that message was not sent, the simulator restarts from the beginning of the round.

Simulating semi-honest corruption requests: Upon receiving from Adv a semi-honest corruption request for party P_i,

1. If $i \notin \mathcal{D}_{\mathsf{SH}}$, ignore the corruption request.
2. Sim initialises live-ctr$_i \leftarrow 0$
3. Send a semi-honest corruption request for party P_i to $\mathcal{W}^{\mathcal{D},\mathsf{Leak},B}_{\mathsf{dist\text{-}graph\text{-}info}}(\mathcal{F}_{\mathsf{BC}})$, and waits to receive $\mathcal{N}_G(i)$
4. Remove i from $\mathcal{D}_{\mathsf{H}}$, and add i to $\mathcal{D}_{\mathsf{SH}}$
5. For $j \in \mathcal{D}_{\mathsf{H}}$:

– add (i, j) and (j, i) to E_{flooding} // update graph G_{flooding}
– add i to $\mathsf{in}(j)$; add j to $\mathsf{in}(i)$ // update incoming adjacency list in

6. For $(u, v) \in [n]^2$:
   If $\mathsf{called\text{-}oracle}[u, v]$ and $v \in \mathcal{N}_G(i)$:
   $\mathsf{oracle}[u, v] \leftarrow \mathsf{oracle}[u, v] - 1$

Simulating fail-stop corruption requests: Upon receiving from Adv a fail-stop corruption request for party P_i

1. If $i \notin \mathcal{D}_{\mathsf{SH}}$, ignore the corruption request.
2. Send a fail-stop corruption request for party P_i to $\mathcal{W}^{\mathcal{D},\mathsf{Leak},B}_{\mathsf{dist\text{-}graph\text{-}info}}(\mathcal{F}_{\mathsf{BC}})$, and waits to receive $\mathcal{N}_G(i)$
3. Remove i from $\mathcal{D}_{\mathsf{SH}}$, and add i to $\mathcal{D}_{\mathsf{FS}}$
4. Atomically invoke subroutine $\mathsf{update\text{-}graph}(R, i, G_{\mathsf{flooding}})$, where R is the index of the last round whose communication phase has concluded

Simulating an even round:

1. For each $i \in [n] \setminus \mathcal{D}_{\mathsf{H}}$:
   (a) If $i \in \mathcal{D}_{\mathsf{SH}} \setminus \mathcal{D}_{\mathsf{LD}}$: // This excludes crashed and low-degree parties
      – Reset $\mathsf{live\text{-}ctr}_i \leftarrow 0$
      – For $j \in \mathcal{N}_G(i)$:
        Send $(i, j, 0)$ to $\mathcal{W}^{\mathcal{D},\mathsf{Leak},B}_{\mathsf{dist\text{-}graph\text{-}info}}(\mathcal{F}_{\mathsf{BC}})$ (on behalf of P_i)
   (b) For each message of the form $(i, _, 0)$ received from $\mathcal{W}^{\mathcal{D},\mathsf{Leak},B}_{\mathsf{dist\text{-}graph\text{-}info}}(\mathcal{F}_{\mathsf{BC}})$ (on behalf of P_i), increment $\mathsf{live\text{-}ctr}_i$
   (c) If $\mathsf{live\text{-}ctr}_i < D$ and $i \notin \mathcal{D}_{\mathsf{LD}}$: // P_i must now "self-crash"
      – add i to $\mathcal{D}_{\mathsf{LD}}$
      – atomically invoke subroutine $\mathsf{update\text{-}graph}(R, i, G_{\mathsf{flooding}})$, where R is the index of the last round whose communication phase has concluded

Simulating a round of the form $2((u-1) + n(v-1) + n^2(r-1)) + 1$***, with*** $(u, v, r) \in [n] \times [n] \times [t+3]$***:***

1. If $((i = u) \vee (u \in \mathcal{N}_G(i))) \wedge \mathsf{high\text{-}degree}_i$, Sim writes $(\mathsf{out}_i, \mathsf{sid}_{u,r})$ on Adv's input tape, as if intended from P_i to $\mathcal{F}_{\mathsf{star\text{-}SFE}}(\mathsf{condOR})$

Simulating a round of the form $2((u-1) + n^2 r) + 1$***, with*** $(u, r) \in [n] \times [t+3]$***:***

1. Compute $d \leftarrow \mathsf{distance}(i^\star, i)$
2. If $r < d$ or $\neg\mathsf{high\text{-}degree}_i$:
   Sim writes $(\mathsf{out}, \perp, \mathsf{sid}_{u,r})$ on Adv's input tape, as if sent from $\mathcal{F}_{\mathsf{star\text{-}SFE}}(\mathsf{condOR})$ to P_i, then updates $\mathsf{out}_i \leftarrow \perp$
3. Else:
   Sim writes $(\mathsf{out}, m, \mathsf{sid}_{u,r})$ on Adv's input tape, as if sent from $\mathcal{F}_{\mathsf{star\text{-}SFE}}(\mathsf{condOR})$ to P_i, then updates $\mathsf{out}_i \leftarrow m$

[a] *i.e.* without advancing the state of Adv; in particular it does not process any more corruption requests before finishing this subroutine.

Fig. 6: Simulator for the topology-hiding broadcast protocol of Figure 5.

Subroutine update-graph

update-graph$(R, i, G_{\text{flooding}})$:
// Remove from G_{flooding} every edge adjacent to i which has not already been used to send the message by the end of the communication phase of round R.
Match R with:

| $2((u-1)+n(v-1)+n^2(r-1))+b$, where $(u,v,r,b) \in [n] \times [n] \times [t+3] \times [0,1]$:
  Compute $d \leftarrow \text{distance}(i^\star, i)$
  For $j \in [V]$:
    If $((i,j) \in E_{\text{flooding}}) \wedge ((d,i,j) \not\preceq_{\text{lex}} (r,u,v))$, remove (i,j) from E_{flooding}
    If $((j,i) \in E_{\text{flooding}}) \wedge ((d,j,i) \not\preceq_{\text{lex}} (r,u,v))$, remove (i,j) from E_{flooding}

| $2((u-1)+n^2 r)+b$, where $(u,r,b) \in [n] \times [t+3] \times [0,1]$:
  Compute $d \leftarrow \text{distance}(i^\star, i)$
  If $(d,i) \not\preceq_{\text{lex}} (r,u)$, remove all edges incident to i

Fig. 7: Subroutine for updating the current flooding prediction graph after a party crashed.

Oracle $\mathcal{O}_{\text{predicted-flooding-trace}}$

The stateful $\mathcal{O}_{\text{predicted-flooding-trace}}$ initially receives the graph G from $\mathcal{F}_{\text{graph}}$, defines the predicted flooding trace graph $G_{\text{predicted-flooding-trace}} := G$, then processes queries from the simulator as follows.

Graph update queries. On input a party index $i \in [n]$ and a round number $R \in [2n(n+1)(t+3)]$ from the simulator, indicating that party P_i was crashed by a fail-stop corruption (or self-crashed due to having low degree) between its communication phase of round $R-1$ and its communication phase of round R (and so, P_i crashed *before* being able to send a message in round R), $\mathcal{O}_{\text{predicted-flooding-trace}}$ atomically runs the subroutine update-graph then returns a ready message to the simulator (indicating it is ready to process more queries).

Distance queries. On input a pair of parties $(i,j) \in [n]^2$, $\mathcal{O}_{\text{predicted-flooding-trace}}$ verifies that it has not received a graph update query with respect to j. If it has not received such a request, it returns $\text{dist}_{G_{\text{predicted-flooding-trace}}}(i,j)$ (with the convention that this is $\perp$ if there is no finite path from i to j), and otherwise it returns $\perp$.

Fig. 8: Distance oracle in the "predicted flooding trace graph"

Subroutine Distance to broadcaster in the current flooding prediction graph

// This is a subroutine for "main" program Sim (Figure 6); all variables are defined therein.

distance$(i^\star, i)$:

1. If $\text{live-ctr}_i < D$:
   Output $\perp$ // Case #1
2. If $i = i^\star$:
   Output 0 // Case #2
3. If $\text{in}(i) \cap \mathcal{D}_{\mathsf{H}} = \varnothing$:
   Output $\min\{\text{distance}(i^\star, j) \colon j \in \text{in}(i)\} + 1$; if this recursion reaches depth n, output $\perp$ instead // Case #3

4. If $i^\star \in \mathsf{in}(i)$:
   Output 1 // Case #4
5. Compute the closed neighbourhood $\mathcal{S}_{\mathsf{reachable}} \subseteq [n]$ of $i^\star$ in the transitive closure of G_{flooding}
6. If $\mathcal{S}_{\mathsf{reachable}} \cap \mathcal{D}_{\mathsf{H}} = \varnothing$:
   Output $\mathsf{dist}_{G_{\mathsf{flooding}}}(i^\star, i)$
   // Note that this can be $+\infty$ // Case #5
7. Compute $d = \mathsf{dist}_{G_{\mathsf{flooding}}}(i^\star, \mathcal{D}_{\mathsf{H}})$ // d is well defined because $\mathcal{S}_{\mathsf{reachable}} \cap \mathcal{D}_{\mathsf{H}} \neq \varnothing$
8. If $\mathsf{dist}_{G_{\mathsf{flooding}}}(i^\star, i) \leq d$:
   Output $\mathsf{dist}_{G_{\mathsf{flooding}}}(i^\star, i)$ // Case #6
9. Compute N_d the set of parties at distance exactly d from $i^\star$ in G_{flooding}
10. If $N_d \cap \mathsf{in}_{G_{\mathsf{flooding}}}(i) \neq \varnothing$:
    Output $d+1$ // Case #7
11. If $|\mathsf{in}(i) \cap \mathcal{D}_{\mathsf{H}}| \geq \frac{1+\varepsilon}{8}np$: // P_i still has at least $\frac{1+\varepsilon}{8}np$ honest neighbours
    Output $d+2$ // Case #8
12. For every $u \in (N_d \cap \mathcal{D}_H)$, if $\neg\mathsf{called\text{-}oracle}[u, i]$:
    - Update $\mathsf{called\text{-}oracle}[u, i] \leftarrow 1$
    - Sim sends a leakage request to $\mathcal{W}^{\mathcal{D},\mathsf{Leak},B}_{\mathsf{dist\text{-}graph\text{-}info}}(\mathcal{F}_{\mathsf{BC}})$ with the message $(u, \mathsf{in}(i) \cap \mathcal{D}_{\mathsf{H}})$ (signifying it wishes to learn the number of edges between u and the parties in $\mathsf{in}(i) \cap \mathcal{D}_{\mathsf{H}}$), and waits to receive an answer, which it stores in $\mathsf{oracle}[u, i]$
13. If there is a $u \in (N_d \cap \mathcal{D}_H)$, such that $\mathsf{oracle}[u, i] > 0$:
    // Note that $\mathsf{called\text{-}oracle}[u, i] = 1$
    Output $d+2$ // Case #9
14. Output $d+3$ // Case #10

Fig. 9: Ideal-world subroutine for determining distance in the current flooding prediction graph.

Let T be a polynomial upper bound on the number of times the simulator calls the subroutine $\mathsf{distance}$, specified in Figure 9 (one can easily derive an explicit, albeit grotesque, bound on T as a function of n and t). We proceed with a hybrid argument:

$\mathcal{H}_0$: This is the real world, where the parties interact with $\mathcal{F}_{\mathsf{star\text{-}SFE}}(\mathsf{condOR})$.

$\mathcal{H}_1$: The environment now interacts with Sim, except that Sim's $\mathsf{distance}$ subroutine is replaced with the oracle call to $\mathcal{O}_{\mathsf{predicted\text{-}flooding\text{-}trace}}$ (Figure 8), which has full knowledge of the initial graph G. Whenever the adversary issues a crash request or whenever a simulated party self-crashes, Sim sends the corresponding corruption request to $\mathcal{O}_{\mathsf{predicted\text{-}flooding\text{-}trace}}$ (which then updates its internal state accordingly). With perfect knowledge of the graph and based on the timing of the crashes, $\mathcal{O}_{\mathsf{predicted\text{-}flooding\text{-}trace}}$ can keep track of the "predicted flooding trace" graph of the flooding procedure (*i.e.* it knows at any given point in time which edges have been used to send the message).

$\mathcal{H}_{2,t}$: For $t \in [0, T]$, hybrid $\mathcal{H}_{2,t}$ is defined as having Sim interact with $\mathcal{W}^{\mathcal{D},\mathsf{Leak},+\infty}_{\mathsf{dist\text{-}graph\text{-}info}}$ for the first t calls to the subroutine $\mathsf{distance}$, and then replacing the subsequent calls to $\mathsf{distance}$ with calls to the oracle $\mathcal{O}_{\mathsf{predicted\text{-}flooding\text{-}trace}}$.

$\mathcal{H}_3$: The environment now interacts with Sim, except that Sim interacts with $\mathcal{W}^{\mathcal{D},\mathsf{Leak},+\infty}_{\mathsf{dist\text{-}graph\text{-}info}}$ (which is the same as $\mathcal{W}^{\mathcal{D},\mathsf{Leak},B}_{\mathsf{dist\text{-}graph\text{-}info}}$ except that it does not bound the number of calls to the leakage oracle Leak).

$\mathcal{H}_4$: This is the ideal world, where the environment interacts with Sim interacting with $\mathcal{W}^{\mathcal{D},\mathsf{Leak},B}_{\mathsf{dist\text{-}graph\text{-}info}}(\mathcal{F}_{\mathsf{BC}})$.

We are now ready to prove indistinguishability of these hybrids.

1. **Hybrids $\mathcal{H}_0$ and $\mathcal{H}_1$ are perfectly indistinguishable.** This is because the output of each instance of "neighbourhood MPC" (namely, $\perp$ or m) depends only on the original graph G and

on the timing of crashes. $\mathcal{O}_{\text{predicted-flooding-trace}}$ has perfect knowledge of this information and uses it to correctly determine these outputs.

2. **Hybrids $\mathcal{H}_1$ and $\mathcal{H}_{2,0}$ are perfectly indistinguishable.** This is because they are code-equivalent.

3. **For $t \in [0, T-1]$, hybrids $\mathcal{H}_{2,t}$ and $\mathcal{H}_{2,t+1}$ are statistically indistinguishable.** We prove this in Lemma 18.

4. **Hybrids $\mathcal{H}_{2,T}$ and $\mathcal{H}_3$ are perfectly indistinguishable.** This is because they are code-equivalent, because T is an upper bound on the number of calls Sim can make to the subroutine distance.

5. **Hybrids $\mathcal{H}_3$ and $\mathcal{H}_4$ are statistically indistinguishable.** Conditioned on Sim not making more than $B = p^{-2}\log^2 n$ leakage requests to the wrapper functionality, the hybrids are identically distributed. It therefore suffices to show that there is a negligible probability that Sim will exceed this bound, which is guaranteed by Lemma 17.

By composing these polynomially many hybrids, the real and ideal worlds are statistically indistinguishable. □

6.1.2 Main Security Arguments

Bounding the number of oracle calls.

Lemma 17 (Bound on the oracle calls made by Sim). *Let* Adv *be an adaptive, fail-stop adversary that interacts with parties $P_1, \ldots, P_n$ running protocol Π_{BC} (Figure 5) in the $(\mathcal{F}_{\text{graph}}^{\mathcal{D}}, \mathcal{F}_{\text{star-SFE}}(\mathsf{condOR}))$-hybrid model. Let* Sim *be the simulator of Figure 6, which interacts with $\mathcal{W}_{\text{dist-graph-info}}^{\mathcal{D},\mathsf{Leak},+\infty}$ (the bound $+\infty$ means the wrapper does not bound the number of leakage queries it processes).*

The event that Sim *makes at least $p^{-2}\log^2 n$ leakage queries to $\mathcal{W}_{\text{dist-graph-info}}^{\mathcal{D},\mathsf{Leak},+\infty}$ is negligible.*[9]

Proof. Observe (by inspection of Figure 9) that the oracle is never called twice on the same pair (u, i) (where $u, i \in [n]$). Moreover, it only calls the oracle on such a pair if at that point in time (which defines the current flooding prediction graph G_{flooding}):

- i still has $\frac{1+\varepsilon}{2}np$ live neighbours (otherwise distance would have terminated on line 1, before making the oracle call of line 11) but fewer than $\frac{1+\varepsilon}{8}np$ honest neighbours in G_{flooding} (otherwise distance would have terminated on line 10);
- u is not a neighbour of any of i's corrupted neighbours in G_{flooding} (otherwise distance would have terminated on line 9).

We now show there is a negligible probability of the simulator calling the oracle on more than kk' different pairs (u, i), where $k = p^{-1}\log^{1/2} n$ and $k' = p^{-1}\log^{3/2} n$. The only sources of randomness are the coins r_S used by Sim and the coins r_G used to generate the random graph G (note that the oracle itself is deterministic). Fix the randomness (r_S, r_G) and let $\{(u_q, i_q) \colon q \in \mathcal{Q}\}$ the set of all queries made by the simulator before it terminates.

1. *Claim: there is at most a negligible fraction of the coins for which $|\{i_q \colon q \in \mathcal{Q}\}| > k$.*

   Let $\mathcal{H}$ be the set of all parties which have not been corrupted by the end of the protocol. The induced subgraph $G[\mathcal{H} \cup \{i_q \colon q \in \mathcal{Q}\}]$ must have size at least $(1-\varepsilon)$ (because the adversary Adv can only corrupt up to $t < \varepsilon n$ parties), and contains at least $|\{i_q \colon q \in \mathcal{Q}\}|$ nodes with degree at most $\frac{1+\varepsilon}{8}np + k$ (because if party i_q still has more than $\frac{1+\varepsilon}{8}np$ honest neighbours by the end of the protocol, then in particular it would also have had more than $\frac{1+\varepsilon}{8}np$ when it

[9] More precisely, this event is independent of Sim's own randomness tape and occurs with negligible probability over the randomness used to generate the Erdös–Rényi random graph.

was queried, which is a contradiction). Since asymptotically $\frac{1+\varepsilon}{8}np + k < \frac{1+\varepsilon}{7}np$, we can invoke Lemma 22, which implies there is only a negligible fraction of the choice of the coins r_G such that $|\{i_q \colon q \in \mathcal{Q}\}| > k$.

2. *Claim: there is at most a negligible fraction of the coins for which* $\exists i \in \{i_q \colon q \in \mathcal{Q}\}, |\{(u_q, i) \colon q \in \mathcal{Q}\}| > k'$.

   Let $v = \arg\max_{i \in \{i_q \colon q \in \mathcal{Q}\}} |\{(u_q, i) \colon q \in \mathcal{Q}\}|$ (*i.e.* v is one of the nodes for which the simulator has the made the most calls). Denote ℓ the number of queries of the form $(_, v)$, and let us enumerate these queries $(u_1, v), \dots, (u_\ell, v)$ in the order they were made. Let $\mathcal{Z}_j$ (resp. $\mathcal{Z}_j$) be the set of live corrupted (resp. honest) neighbours of v at the time query (u_j, v) was made. Observe that:
   - $\mathcal{H}_1 \supseteq \cdots \supseteq \mathcal{H}_\ell$ (because a corrupted party cannot later become honest again)
   - For every $j \in [\ell - 1]$, $\mathcal{Z}_{j+1} \cap \mathcal{Z}_j \supseteq \mathcal{Z}_{j+1} \cap \overline{\mathcal{H}}_j$. Indeed, any node $x \in \mathcal{Z}_{j+1} \cap \overline{\mathcal{H}}_j$ was still live at the time query (u_{j+1}, v) was made (because $x \in \mathcal{Z}_{j+1}$), so it must have been live when query (u_j, v) was made. The set of live neighbours of v at that time was exactly $\mathcal{Z}_j \cap \overline{\mathcal{H}}_j$. But since $x \in \overline{\mathcal{H}}_j$, this implies that $x \in \mathcal{Z}_j$.

   By combining these facts:

$$\begin{aligned}
\bigcap_{j=1}^{\ell} \mathcal{Z}_j &= \left(\bigcap_{j=3}^{\ell} \mathcal{Z}_j\right) \cap \underbrace{(\mathcal{Z}_2 \cap \mathcal{Z}_1)}_{\supseteq \mathcal{Z}_2 \cap \overline{\mathcal{H}}_1} \\
&\supseteq \left(\bigcap_{j=2}^{\ell} \mathcal{Z}_j\right) \cap \overline{\mathcal{H}}_1 \\
&\supseteq \left(\bigcap_{j=4}^{\ell} \mathcal{Z}_j\right) \cap \underbrace{(\mathcal{Z}_3 \cap \mathcal{Z}_2)}_{\supseteq \mathcal{Z}_3 \cap \overline{\mathcal{H}}_2} \cap \overline{\mathcal{H}}_1 \\
&\dots \\
&\supseteq Z_\ell \cap \underbrace{\overline{\mathcal{H}}_{\ell-1} \cap \cdots \cap \overline{\mathcal{H}}_1}_{= \overline{\mathcal{H}}_1}
\end{aligned}$$

   And so $|\bigcap_{j=1}^{\ell} \mathcal{Z}_j| \geq |\mathcal{Z}_\ell \cap \overline{\mathcal{H}}_1| \geq |\mathcal{Z}_\ell| - |\mathcal{H}_1| \geq \frac{1+\varepsilon}{2}np - \frac{1+\varepsilon}{8}np \geq \frac{1+\varepsilon}{4}np$.

   Now consider the induced subgraph $G[\left(\bigcap_{j=1}^{\ell} \mathcal{Z}_j\right) \cup \{u_1, \dots, u_\ell\} \cup \{v\}]$. On the one hand, v has at least $\frac{1+\varepsilon}{4}np$ neighbours in this induced subgraph (by definition of the $\mathcal{Z}_j$, any element of $\bigcap_{j=1}^{\ell} \mathcal{Z}_j$). At the same time, none of the $(u_j)_{j \in [\ell]}$ can be in the 2-neighbourhood of v.[10] By Lemma 24, such an induced subgraph can only exist for a negligible fraction of the possible coins r_G used to generate the random graph.

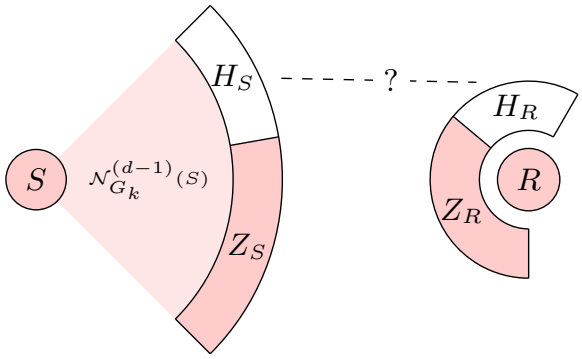
By combining the two claims, there is only a negligible probability that the simulator makes more than $kk' = p^{-2} \log^2 n$ calls to the oracle. Therefore, hybrids $\mathcal{H}_1$ and $\mathcal{H}_2$ are indeed statistically indistinguishable. □

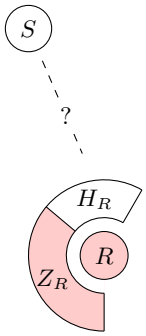
Correctness of the distance ***subroutine.*** The main idea behind the subroutine distance is that the simulator, based on the neighbourhoods of the corrupted parties, has perfect knowledge of the "predicted flooding trace graph" G_{flooding} (which is the one the oracle G_{flooding} maintains) up to some distance d away from the sender. This is represented in Figure 10. The goal of distance is therefore to determine the distance of each corrupted party to the set H_S of the honest parties at distance d from S.

Lemma 18. *With the notations of the proof of Theorem 16, the hybrids $\mathcal{H}_{2,t}$ and $\mathcal{H}_{2,t+1}$ are statistically indistinguishable for all $t \in [0, T-1]$.*

[10] Indeed, suppose there were a party u_j in the 2-neighbourhood of v. Since u_j and v cannot be neighbours (or the query (u_j, v) would never have been made because of line 3 in Figure 9). So they would have to be at distance exactly 2, *i.e.* they have a common neighbour in $S \coloneqq \left(\bigcap_{j=1}^{\ell} \mathcal{Z}_j\right) \cup \{u_i \colon i \in [\ell] \setminus \{j\}\}$. Except u_j cannot have a neighbour in $\mathcal{Z}_j \supseteq S$ (because of line 9 in Figure 9) while v cannot have a neighbour in $\{u_i \colon i \in [\ell] \setminus \{j\}\} \supseteq S$. This is a contradiction.

(a) The d-neighbourhood of S in G_k is partitioned as $\mathcal{N}_{G_k}^{(d-1)}(S)$ (the nodes at distance at most $d-1$ from S), H_S (the honest parties at distance exactly d from S), Z_S (the corrupted parties at distance exactly d from S). The (1-)neighbourhood of R is partitioned as H_R (its honest neighbours) and Z_R (its corrupted neighbours).





(b) The case where the sender is honest is the special case $d = 0$ (the adversary only has knowledge of the broadcaster, not its neighbourhood) and $H_S = \{S\}$.

Fig. 10: A visual representation of the simulator's partial knowledge of the nodes who have received the message, up to some distance $d \geq 0$ away from the broadcaster S.

Proof. The only difference between these two hybrids is whether the simulator calls the oracle $\mathcal{O}_{\text{predicted-flooding-trace}}$ or runs the subroutine distance for its tth call. Let us show that there is only a negligible probability that their answers differ. As highlighted in Figure 9, there are ten different cases for what distance can output. In cases $1-7$ and 9, the two oracles' outputs are trivially the same, but cases 8 and 10 require non-trivial analysis.

- *Case #1*: If $\text{live-ctr}_i < D$, Sim must have sent a (self-)corruption request for i to $\mathcal{O}_{\text{predicted-flooding-trace}}$ in H_t, so both world's output must be $\perp$.

- *Case #2*: By definition of distance, $\text{dist}(i,i)=0$.

- *Case #3*: By definition of distance, if $i \neq i^\star$ then i is exactly one hop further from $i^\star$ than i's closest neighbour is. If distance goes into an infinite loop, it must mean there is no path from $i^\star$ to i (as this implies that i is in a connected component containing only corrupted parties, and which also does not contain $i^\star$).

- *Case #4*: If i and $i^\star$ are neighbours they are at distance exactly 1.

- *Case #5, #6, and #7*: Sim's knowledge of the graph extends to a distance far enough away from $i^\star$ that it knows i's distance exactly.

- *Case #8*: By Lemma 17, the simulator has only made $p^{-2}\log^2 n$ leakage queries to the wrapper (via subroutine distance) so far, and each one returned an index in $[n]$. Therefore the simulator got at most $p^{-2}\log^3 n = o(np)$ bits of leakage. Let us now argue there is only a negligible probability that i does not have a common neighbour with $\mathcal{D}_H$ (the set of honest parties at distance d from $i^\star$, which is non-empty by definition of d).

  Consider the game of Core Lemma 1, with the following adversary $\mathcal{A}$ interacting with $\mathcal{O}_{\text{neigh}}$: Adv starts by sampling randomness $r \xleftarrow{\$} \{0,1\}^{p^{-2}\log^3 n}$, then runs a copy of Sim up to the tth query to distance, using $\mathcal{O}_{\text{neigh}}$ as a substitute for the corruption request Sim makes to $\mathcal{W}^{\mathcal{D},\text{Leak},B}_{\text{dist-graph-info}}$, and the randomness r as a substitute for the leakage queries to $\mathcal{W}^{\mathcal{D},\text{Leak},B}_{\text{dist-graph-info}}$. Let $B \leftarrow p^{-2}\log^3 n$, and note that $\frac{np}{8} \in \frac{\omega(\log n)+\log_2(e)\cdot B}{p}$. By Core Lemma 1, the probability is at most $n^{-\omega(1)}/2^B$ that $\mathcal{A}$ succeeds in causing a pair (u,v) with the following properties
  1. u has not been queried
  2. at least $\frac{np}{8}$ neighbours of v have not been queried
  3. u and v do not have an unqueried neighbour.

By a union bound over the choice of $r \in \{0,1\}^B$, there is a negligible probability that there exists a "leakage bitstring" which would cause $\mathcal{A}$ to cause such a pair (u, v) to exist. It follows that there is at most a negligible probability (over the randomness of the Erdös–Rényi graph and the randomness tape of Sim) that Sim can ever cause a corrupted party with at least $\frac{np}{8}$ honest neighbours to not have a common neighbour with each and every honest party. In particular, $d(i, \mathcal{D}_H) \leq 2$ with all but negligible probability.

We have shown that $\mathsf{dist}(i, \mathcal{D}_H) \leq 2$ (and therefore $\mathsf{dist}(i, i^\star) \leq d + 2$) with all but negligible probability. Note that because the condition of line 10 did not trigger (corresponding to case #10), $\mathsf{dist}(i, i^\star) > d + 1$, and we can conclude that $\mathsf{dist}(i, i^\star) = d + 2$ with all but negligible probability.

– *Case #9*: i cannot be at distance $\leq d + 1$, as otherwise the subroutine would have terminated before line 13; this means that it must be at distance exactly 2, because it has a neighbour at distance $d + 1$ (this is given by the leakage).

– *Case #10*: By an analogous reduction to Case 8, but using the game of Lemma 22 instead of that of Core Lemma 1, there is a negligible probability that any honest party has fewer than $\frac{1+\varepsilon}{2}$ honest neighbours, and so in particular fewer than $\frac{1+\varepsilon}{2}$ live neighbours. Therefore, by Lemma 23, there is only a negligible probability that i and $i^\star$ are not within $d + 3$ hops.

□

6.2 Instantiating the Hybrid Model

Having the parties perform a "neighbourhood MPC" has been a standard technique of topology-hiding protocols since their inception [MOR15]. If a party u wishes to perform secure computation with its neighbourhood, it issues one-time pseudonyms to its neighbours (so that corrupted parties cannot learn the identities of the honest participants), and simulates "virtual neighbours" in order to pad up its neighbourhood up to some public upper bound (in order to not reveal the degree of u). Note that the simulator knows which corrupted parties participate in which neighbourhood MPCs, as it knows exactly which corrupted parties are neighbours of u.
This solution removes all the topology-hiding aspects of the problem, and we are left to design an adaptively secure fail-stop MPC protocol in a star topology (with a public number of branches). Because we only need to realise functionalities with solitary output to the center u, this is a straightforward task. The parties can use an adaptively secure key-agreement protocol to establish a complete overlay network of secure point-to-point channels (recall that the adversary already knew if two corrupted parties were participating in the same computation, so establishing such channels is not a problem), with the center u acting as a router, passing on unintelligible key-agreement messages.
Once this complete overlay network is established, the parties can use an adaptively secure MPC protocol (*e.g.* [CLOS02] that is based on adaptively secure oblivious transfer) to compute the desired function. If the center ever crashes, there is no need to continue because none of the other parties obtains an output anyway (we only deal with solitary output to the center). If one of the other parties crashes, u can simply inform all of its neighbours, and the computation can then be aborted and restarted (note that this can only happens up to $t + 1$ times, so we can set a fixed polynomial upper bound on the number of rounds which should be reserved for each neighbourhood MPC, while still ensuring guaranteed output delivery). Since only corrupted parties can crash (and the simulator therefore knows which neighbourhood MPCs are affected), restarting in this way does not violate the topology-hiding requirement.

Corollary 19 (Distributional Topology-Hiding Computation with Bounded Leakage). *Let $\varepsilon \in (0, 1)$ be a constant, let $t \leq \varepsilon n$, let $p \leftarrow n^{-1/3} \log^2 n$, let $\mathcal{D}$ is the Erdös–Rényi distribution $\mathsf{ER}(n, p)$, let $B \leftarrow p^{-2} \log^2 n$, and let Leak is the function which, on input $(S_1, S_2) \in (2^{[n]})^2$, outputs the number of edges in $S_1 \times S_2$.*
Assuming the existence of adaptive oblivious transfer, there exists a protocol which securely realises $\mathcal{W}^{\mathcal{D},\mathsf{Leak},B}_{\mathsf{dist\text{-}graph\text{-}info}}(\mathcal{F}_{\mathsf{BC}})$ in the $\mathcal{F}^{\mathcal{D}}_{\mathsf{graph}}$-hybrid model.

7 Core Lemmata

In this section we establish a few facts about Erdös–Rényi graphs which form the basis for (the design and analysis of) our protocol of Section 6, which is our main result.

7.1 Distributed Sampling of an Erdös–Rényi Graph

In this section, we recall for completeness the known fact [CGO12b, CGO12a, CCG+15] that n parties can securely sample an Erdös–Rényi graph of parameters (n, p) in the hidden channel model with locality $\sim np$. The protocol consists in each party pinging parties uniformly at random, each with some well-chosen probability p': the Erdös–Rényi graph output by the protocol is the one containing all edges (u, v) where u and v pinged each other.
Prior works [CGO12b, CGO12a, CCG+15] have sometimes considered a variant of $\Pi_{\mathsf{ER}(n,p)}$ wherein we add the edge (u, v) to the Erdös–Rényi graph if and only if *both* u and v have pinged each other. Because we only consider semi-honest and fail-stop adversaries however, we can safely switch to adding the edge (u, v) to the Erdös–Rényi graph if and only if *at least one* of u or v has pinged the other. While this does not change the asymptotic locality of the protocol, it provides the elegant feature that setting up the Erdös–Rényi graph using $\Pi_{\mathsf{ER}(n,p)}$ "comes for free" in terms of locality, as every communication channel which was used ends up being added to the graph.

Functionality Distributed Sampling of an Erdös–Rényi Graph $\mathcal{F}_{\mathsf{ER}(p)}$

Parameters: The ideal functionality $\mathcal{F}_{\mathsf{ER}(p)}$ is parameterised by a number of parties n and an Erdös–Rényi parameter $p \in (0, 1)$.

$\mathcal{F}_{\mathsf{ER}(p)}$ interacts with the N parties $P_1, \ldots, P_n$ in the following manner.

Input: Wait to receive $(\texttt{ready}, i)$ from each party P_i.

Output: Run $G = ([n], E) \hookleftarrow \mathsf{ER}(n, p)$ and parse E to retrieve the list of neighbourhoods $L_1, \ldots, L_N$; Output L_i to each party P_i $(1 \leq i \leq n)$.

Fig. 11: Ideal functionality $\mathcal{F}_{\mathsf{ER}(p)}$ for the distributed generation of an Erdös–Rényi graph, each party obtaining its neighbourhood.

Protocol Distributed Sampling of an Erdös–Rényi Graph $\Pi_{\mathsf{ER}(p)}$

Parties: $P_1, \ldots, P_n$

Parameters: The protocol is parameterised with a number of parties n and an Erdös–Rényi graph parameter p (which may be a function of n).

The Protocol: Each party P_i $(i \in [n])$ does the following:

- ***Initialisation Phase:***
  1. Initialise neighbourhood list to $L_i \coloneqq 0^n$.
  2. Sample $n - 1$ *i.i.d.* Bernoulli random variables $(X_{i,j})_{j \in [n] \setminus \{i\}} \hookleftarrow \mathsf{Ber}(1 - \sqrt{1-p})$.
  3. For $j \in [n] \setminus \{i\}$, initialise $b_{i,j} \coloneqq 0$.
- ***Communication Phase:***
  3. For $j \in [n] \setminus \{i\}$, if $X_{i,j} = 1$ then send a bit to party P_j .
  4. Wait to receive any message in this round, and upon reception of a message from P_j $(j \in [n] \setminus \{0, 1\})$ set $b_{i,j} \leftarrow 1$.
- ***Output Phase:***
  5. For $j \in [n] \setminus \{i\}$, if $X_{i,j} = 1$ or $b_{i,j} = 1$ then set $L_i[j] \leftarrow 1$.
  6. Output L_i .

Fig. 12: Protocol $\Pi_{\mathsf{ER}(p)}$ for securely distributing the generation of an Erdös–Rényi graph, each party obtaining its neighbourhood, in the presence of an adaptive fail-stop adversary.

Lemma 20 (An Erdös–Rényi Graph Can be Securely Sampled with Low-Locality). *Let n be a number of parties, let $p \in (0,1)$ (p may be a function of N), and let $t = \mathcal{O}(n)$. In the hidden channel model, protocol $\Pi_{\mathsf{ER}(p)}$ securely realises $\mathcal{F}_{\mathsf{ER}(p)}$—parameterised by (n,p)—with perfect security in the presence of an adaptive fail-stop adversary corrupting at most t parties.*
Furthermore the locality of this protocol follows the law of the maximum degree of an Erdös–Rényi graph of parameters (n,p), and in particular for any $\delta > 0$ the probability that $\Pi_{\mathsf{ER}(p)}$ has locality (at most) $np \cdot (1+\delta)$ is upper bounded by $n \cdot \exp(-\delta^2 \cdot np/3)$.

Proof. There are two claims we need to prove: the first is that $\Pi_{\mathsf{ER}(p)}$ is secure, and the second is that it indeed has the above locality. As a sanity check however, let us first establish that the protocol is "correct", *i.e.* that it indeed samples an Erdös–Rényi graph with parameters (n,p) in a distributed way.

- *"Correctness".* Let $p' \coloneqq 1 - \sqrt{1-p}$. Observe that in the final graph there is an edge between parties P_i and P_j (*i.e.* $L_i[j] = L_j[i] = 1$) if and only if $X_{i,j} = 1$ or $X_{j,i} = 1$. Since $X_{i,j}$ and $X_{j,i}$ are *i.i.d.* $\mathsf{Ber}(1-\sqrt{1-p})$, this happens with probability $2 \cdot p' \cdot (1-p') + (p')^2 = p$. Hence correctness.
- *Security.* The subgraph induced by the honest parties is information-theoretically independent of the corrupt parties' views.
- *Locality of $\Pi_{\mathsf{ER}(p)}$.* By the arguments from the proof of correctness, the communication graph of the protocol is a random Erdös–Rényi graph of parameters (n,p), and so by definition the communication locality follows the claimed law. For every $i \in [n]$, the degree $\deg_i$ of party P_i in the communication graph therefore follows a binomial distribution of parameters (n,p), of mean $\mu = np$. In particular, by a Chernoff bound, $\forall \delta > 0, \Pr[\deg_i \geq \mu \cdot (1+\delta)] \leq \exp(-\delta^2 \cdot \mu/(2+\delta))$. We then derive the desired bound via union bound on $i \in [n]$.

□

7.2 Structural properties of Erdös–Rényi graphs

7.2.1 An honest party's neighbourhood has a constant fraction of honest parties. In this section, we show that an adversary learning an Erdös–Rényi graph through neighbourhood queries (even adaptively, as a function of the answers to prior queries) cannot learn significantly more neighbours of an unqueried node than it would by issuing queries uniformly at random. The main technical reason for this is that all the edges are independent, so the subgraph induced by unqueried nodes always has full entropy. The only leverage the adversary has is therefore to adaptively chose which neighbourhood *not* to query (*e.g.* because it has, by chance, already identified more of its neighbours than expected), but a union bound shows this advantage to be negligible.

Lemma 21 (Honest parties have high degree). *Let $\mathcal{O}_{\mathsf{neigh}}$ be a stateful oracle with the following properties:*

- *$\mathcal{O}_{\mathsf{neigh}}$ initially holds an input x, internally samples an Erdös–Rényi graph G with parameters (n,p) (this graph is independent of x), and initialises $Q \leftarrow \varnothing$;*
- *on input a query $i \in [n]$, $\mathcal{O}_{\mathsf{neigh}}$ updates $Q \leftarrow Q \cup \{i\}$ then outputs $(\mathcal{N}_G(j))_{j \in Q}, x)$.*

Let $\varepsilon \in (0,1)$, let $t = \omega(\frac{\log n}{p})$ but s.t. *$t < n$, and let $k \leq t$. The probability that an oracle machine making k calls to $\mathcal{O}_{\mathsf{neigh}}$ outputs a pair $(i,S) \in [n] \times 2^{[n]}$ satisfying*

1. *$\mathcal{O}_{\mathsf{neigh}}$ was not queried on i*
2. *$\mathcal{O}_{\mathsf{neigh}}$ was queried on every $j \in S$*
3. *$S \subseteq \mathcal{N}_G(i)$ and $|S| \geq (1+\varepsilon) \cdot tp$.*

is at most $\exp(-\varepsilon^2 \cdot tp/(2+\varepsilon))/n = n^{-\omega(1)}$.

Proof. Let $\mathcal{A}$ be an oracle machine which succeeds in outputting such a valid pair (i, S) with probability q.

- *Reduction to the fixed-i case.* Now consider the alternative game, where the target i is fixed:

  > The adversary succeeds if, after at most t queries to $\mathcal{O}_{\mathsf{neigh}}$ but without querying i, it can output $S \in 2^{[n]}$ such that (1) $\mathcal{O}_{\mathsf{neigh}}$ was queried on every $j \in S$, (2) $S \subseteq \mathcal{N}_G(i)$, and (3) $|S| \geq tp(1+\varepsilon)$.

  By the law of total probabilities, $q = q_1 + \cdots + q_n$, where q_i is the probability $\mathcal{A}$ succeeds with an output of the form $(i, \cdot)$. Let $i^\star \coloneqq \mathsf{argmax}\{q_i \colon i \in [n]\}$, and observe that $q_{i^\star} \geq q/n$. Then (by running $\mathcal{A}$ and aborting if it queries $\mathcal{O}_{\mathsf{neigh}}$ on $i^\star$) there exists an oracle machine which succeeds with probability $q_{i^\star}$ in the fixed-$i^\star$ variant of the game.
- *Analysing the fixed-i setting.* Let $i \in [n]$, and let us upper bound the probability an adversary can win the above fixed-i variant of the game. Because each query to $\mathcal{O}_{\mathsf{neigh}}$ has independently probability p of being in $\mathcal{N}_G(i)$, no matter its strategy, an oracle machine can only succeed in the fixed-i game if it achieves at least $tp(1+\varepsilon)$ successes in t independent Bernoulli trials with the same rate p. By a Chernoff bound (with respect to the binomial distribution $\mathcal{B}(t,p)$), it can therefore only win with probability at most $e^{-\varepsilon^2 \cdot tp/(2+\varepsilon)}$.
- *Conclusion.* $q \leq q_{i^\star}/n \leq e^{-\varepsilon^2 \cdot tp/(2+\varepsilon)}/n = n^{-\omega(1)}$.

□

7.2.2 Almost every node has a constant fraction of honest neighbours. In this section, we show that, with all but negligible probability, an Erdös–Rényi graph does not admit a linear-sized induced subgraph where $\omega(p^{-1})$ nodes have degree $o(np)$. On a technical level, this is because Erdös–Rényi are too dense to admit such a sparse substructure. In our protocol, this means that (w.a.b.n.p.over the generation of the random graph) the adversary can only ever cause $O(p^{-1})$ nodes to not have $\Omega(np)$ honest neighbours, no matter how much leakage about the graph is given to the adversary.

Lemma 22 (Few nodes can ever have few honest neighbours). *Let $\delta, \varepsilon > 0$ such that $\delta < 1 - \varepsilon$. Let $G = (V, E)$ be an Erdös–Rényi random graph of parameters (n, p) with $p = \omega(\frac{\log n}{n})$. Let $k \in \omega(1/p) \cap o(n/\log n)$. There is a negligible (in n) probability that G admits an induced subgraph H of size at least $(1-\varepsilon) \cdot n$ with at least k nodes of degree less than $\delta \cdot np$.*

Proof. Let A be the event whose probability we wish to bound. For all $S \in \binom{[n]}{\leq \varepsilon \cdot np}$ (the choice of the nodes not in H) and all $T \in [n] \setminus \binom{S}{k}$ (the choice of the low degree nodes in H), consider the event:

> $A_{S,T}$: "for all $u \in T$, u has fewer than $\delta \cdot np$ neighbours in $[n] \setminus S$"

and observe that $A = \cup_{S,T} A_{S,T}$. For all $S \in \binom{[n]}{\leq \varepsilon \cdot np}$ and all $T \in \binom{[n] \setminus S}{k}$, and $u \in T$, define event $B_{S,T,u}$ as:

> $B_{S,T,u}$: "u has fewer than $\delta \cdot np - k$ neighbours in $[n] \setminus (S \cup T)$"

and observe that $A_{S,T} \subseteq (\cap_{u \in T} B_{S,T,u})$ (indeed, if a node has fewer than $\delta \cdot np - k$ neighbours in $[n] \setminus (S \cup T)$, it must also have fewer than fewer than $\delta \cdot np$ neighbours in $[n] \setminus S$ as $|T| = k$). For each choice of S, T, the events $\{B_{S,T,u} \colon u \in T\}$ are mutually independent, as they are regarding non-overlapping subsets of the edges of G. Each $B_{S,T,u}$ corresponding to fewer than $\delta \cdot np - k \leq \delta \cdot np$ successes in $n - |S| - k \geq (1 - \varepsilon - \frac{(1-\varepsilon)+\delta}{2}) \cdot n = \frac{(1-\varepsilon)+\delta}{2} n$ [11] *i.i.d.* $\mathsf{Ber}(p)$, $\Pr[B_{S,T,u}] \leq \mathsf{cst}_{\delta,\varepsilon}^{np}$ by a Chernoff bound, where $\mathsf{cst}_{\delta,\varepsilon} \in (0,1)$ is a constant which depends only on δ and ε.

[11] Since $k = o(n)$, for a large enough value of n, $k \leq \frac{(1-\varepsilon)+\delta}{2} n$.

$$
\begin{aligned}
\Pr[A] &= \Pr[\bigcup_{S,T} A_{S,T}] \\
&\leq \sum_S \sum_T \Pr[A_{S,T}] && \text{// By a union bound} \\
&\leq \sum_S \sum_T \Pr[\bigcap_{u\in T} B_{S,T,u}] && \text{// } A_{S,T} \subseteq \textstyle\bigcap_{u\in T} B_{S,T,u} \\
&\leq \sum_S \sum_T \prod_{u\in T} \Pr[B_{S,T,u}] && \text{// By independence of the } B_{S,T,u} \\
&\leq \sum_{i=0}^{\varepsilon n} \sum_{\substack{S \text{ s.t.} \\ |S|=i}} \sum_T \prod_{u\in T} \mathsf{cst}_{\delta,\varepsilon}^{np} && \text{// By a Chernoff bound} \\
&\leq \varepsilon n \cdot 2^n \cdot n^k \cdot (\mathsf{cst}_{\delta,\varepsilon}^{np})^k = n^{-\omega(1)} && \text{// Because } k \in \omega(1/p) \cap o(n/\log n)
\end{aligned}
$$

□

7.2.3 All high-degree induced subgraphs have diameter at most three. In this section, we show that, with all but negligible probability, an Erdös–Rényi graph does not admit an induced subgraph where two nodes with degree $\Omega(np)$ are *not* within three hops. In our protocol, this means that no matter which parties the adversary chooses to crash, nodes with $\Omega(np)$ live neighbours (whether honest or semi-honest) are all guaranteed (w.a.b.n.p.over the randomness of the Erdös–Rényi graph) to be at distance at most three of each other.

Lemma 23 (High-degree nodes are within three hops). *Let $\delta \in (0,1)$. Let $G = (V, E)$ be an Erdös–Rényi random graph of parameters (n, p), with $1 \geq p \geq \sqrt{2\ln n}/\sqrt{\delta n}$. The probability that G admits an induced subgraph G' containing two vertices u and v such that*

1. *both u and v have degree at least $\delta \cdot np$ in G'*
2. *the distance between u and v in G' is more than 3*

is negligible (in n).

Proof. Observe that the event can be re-written as A: "there exist $2 + 2 \cdot \lceil \delta \cdot np \rceil$ distinct nodes $u, u_1, \ldots, u_{\lceil \delta \cdot np \rceil}, v, v_1, \ldots, v_{\lceil \delta \cdot np \rceil}$ such that for every $i \in [\lceil \delta \cdot np \rceil]$, $(u, u_i), (v, v_i) \in E$ but $\{u, u_1, \ldots, u_{\lceil \delta \cdot np \rceil}\} \times \{v, v_1, \ldots, v_{\lceil \delta \cdot np \rceil}\} \cap E = \varnothing$". By a union bound over the choice of these nodes (and using the facts $\binom{a}{b} \leq a^b$ that and that $\ln(1-p) \leq -p - p^2/2$),

$$
\begin{aligned}
\Pr[A] &\leq \binom{n}{2 + 2 \cdot \lceil \delta \cdot np \rceil} \cdot p^{2\lceil \delta \cdot np \rceil} \cdot (1-p)^{\lceil \delta \cdot np \rceil^2} \\
&\leq n^{2+2\cdot\lceil \delta \cdot np \rceil} \cdot p^{2\delta \cdot np} \cdot (1-p)^{(\delta \cdot np)^2} \\
&\leq n^{2+2\cdot\lceil \delta \cdot np \rceil} \cdot p^{2\delta \cdot np} \cdot \exp\left((\delta \cdot np)^2 \cdot (-p - p^2/2)\right) \\
&\leq n^4 \cdot n^{2\delta \cdot np} \cdot p^{2\delta \cdot np} \cdot \exp\left(-\delta \cdot n^2 p^3 - \delta \cdot n^2 p^4/2)\right) \\
&\leq n^4 \cdot \exp\left(\underbrace{2\delta \cdot np \cdot \overbrace{\ln(np)}^{\leq \ln n} - \delta^2 \cdot n^2 p^3}_{\leq 0 \text{ by assumption on } p} - \underbrace{\delta^2 n^2 p^4/2}_{=\omega(\log n)} \right) \\
&= 1/n^{\omega(1)} \ .
\end{aligned}
$$

□

7.2.4 High-degree nodes are within two hops of almost all nodes. In this section, we refine the result of Lemma 23, and show that not only should all nodes with degree $\Omega(np)$ be within three hops of each other, but furthermore that each one should be within *two* hops of all but at most $O(\frac{\log n}{p})$ (in particular, these last nodes need not even have degree $\Omega(np)$). In our protocol, this means that no matter which parties the adversary chooses to crash, nodes with $\Omega(np)$ live neighbours (whether honest or semi-honest) are all guaranteed (w.a.b.n.p.over the randomness of the Erdös–Rényi graph) to be at distance at most two of all but at most $O(\frac{\log n}{p})$ live nodes.

Lemma 24 (High-degree nodes are within two hops of almost all nodes). *Let $\delta \in (0,1)$. Let $G = (V, E)$ be an Erdös–Rényi random graph of parameters (n, p) where $p > 0$ and $p = o(1)$, and let $k = \omega(\frac{\log n}{p})$. The probability that G admits an induced subgraph G' containing a vertex u such that*

1. *u has degree at least $\delta \cdot np$ in G'*
2. *there are at least k nodes in G' whose distance from u in G' is at least 3*

is negligible in n.

Proof. Denote A the event whose probability we wish to bound ("G admits [...] at least 3"). First observe that A is equivalent to the following event B:

B : "There exists $(u, v_1, \ldots, v_{\lceil \delta \cdot np \rceil}, w_1, \ldots, w_k) \in \binom{V}{1+\lceil \delta \cdot np \rceil + k}$ such that:
- ***(v_i are all neighbours of u)*** $\forall i \in [1, \lceil \delta \cdot np \rceil], (u, v_i) \in E$
- ***(none of the w_j are 1-neighbours of u)*** $\forall j \in [k], (u, w_j) \notin E$
- ***(none of the w_j are 2-neighbours of u)*** $\forall i \in [1, \lceil \delta \cdot np \rceil], \forall j \in [k], (v_i, w_j) \notin E$ "

These events are indeed equivalent as:

- if event A holds, choosing $v_1, \ldots, v_{\lceil \delta \cdot np \rceil}$ arbitrarily from the neighbours of u in G' and choosing $w_1, \ldots, w_k$ arbitrarily from the nodes of G' which are not in the 2-neighbourhood (in G') of u constitutes a "witness" for event B;
- if event B holds, the induced subgraph $G' \coloneqq G[u, v_1, \ldots, v_{\lceil \delta \cdot np \rceil}, w_1, \ldots, w_k]$ constitutes a "witness" for event A.

We can therefore analyse the probability of event B. By a union bound (over the choice of $(u, v_1, \ldots, v_{\lceil \delta \cdot np \rceil}, w_1, \ldots, w_k)$), and using the fact all these edges are i.i.d. $\mathsf{Ber}(p)$:

$$\begin{aligned}
\Pr[A] &\leq \binom{n}{1 + \lceil \delta \cdot np \rceil + k} \cdot p^{\lceil \delta \cdot np \rceil} \cdot (1-p)^{k + \lceil \delta \cdot np \rceil k} \\
&\leq \exp[\underbrace{(\log n)(2 + \delta \cdot np + k)}_{= o(\delta np^2 k) \text{ (because } kp = \omega(\log n))} + \underbrace{\delta np \ln(p)}_{\leq 0} + \underbrace{(-kp)}_{\leq 0} - \delta np^2 k] \\
&= 2^{-\omega(np \cdot \log n)} = n^{-\omega(1)}
\end{aligned}$$

□

7.2.5 A party with $\Omega(np)$ honest neighbours has a common neighbour with each and every honest party.

Core Lemma 1 (A party with $\boldsymbol{\Omega(np)}$ honest neighbours has a common neighbour with each and every honest party)**.** *Let $\mathcal{O}_{\mathsf{neigh}}$ be a stateful oracle with the following properties:*

- *$\mathcal{O}_{\mathsf{neigh}}$ initially holds an input x, internally samples an Erdös–Rényi graph G with parameters (n, p) (this graph is independent of x), and initialises $Q \leftarrow \varnothing$;*
- *on input a query $i \in [n]$, $\mathcal{O}_{\mathsf{neigh}}$ updates $Q \leftarrow Q \cup \{i\}$ then outputs $(\mathcal{N}_G(j))_{j \in Q}, x)$.*

Let $B > 0$. Let $\varepsilon \in (0,1)$, let $k \leq n$. Consider an oracle machine which makes k calls to $\mathcal{O}_{\mathsf{neigh}}$. The probability that there exists a pair of distinct nodes $u, v \in [n]$ with the following properties is at most $n^{-\omega(1)}/2^B$:

- *u has not been queried;*
- *at least $\frac{\omega(\log n) + \log_2(e) \cdot B}{p}$ neighbours of v have not been queried;*
- *every common neighbour of u and v has been queried.*

Proof. Let $\mathcal{A}$ be an oracle machine which succeeds with probability q creating such a pair (u, v).

- *Reduction to the fixed-(u, v) case.* Now consider the alternative game, where the targets u and v are fixed:

  The adversary succeeds if it can corrupt all of the common neighbours of u and v while leaving u and at least $\frac{\omega(\log n) + \log_2(e) \cdot B}{p}$ neighbours of v unqueried.

By a union bound, $q \leq \sum_{u \neq v} q_{u,v}$, where $q_{u,v}$ is the probability $\mathcal{A}$ corrupted all of the common neighbours of u and v while leaving u and at least $\frac{\omega(\log n)+\log_2(e)\cdot B}{p}$ neighbours of v unqueried. Let $(u^\star, v^\star) \coloneqq \mathsf{argmax}\{q_{u,v} \colon (u,v) \in [n], u \neq v\}$, and observe that $q_{u^\star,v^\star} \geq \frac{q}{n(n-1)}$. Then (by running $\mathcal{A}$ and aborting if it queries $\mathcal{O}_{\mathsf{neigh}}$ on $u^\star$) there exists an oracle machine which succeeds with probability $q_{u^\star,v^\star}$ in the fixed-$(u^\star, v^\star)$ variant of the game.

– *Analysing the fixed-(u,v) setting.* Let $u, v \in [n]^2$ *s.t.* $u \neq v$. Without loss of generality (as it can only increase the adversary's success probability), we can assume that the adversary starts by querying v, and that it subsequently queries only neighbours of the broadcaster until there are exactly $\lceil \frac{\omega(\log n)+\log_2(e)\cdot B}{p} \rceil$ ones left. Regardless of how the adversary proceeds, the remaining $\lceil \frac{\omega(\log n)+\log_2(e)\cdot B}{p} \rceil$ parties each have probability p of being neighbours of u, independently of all the information available to the adversary. Therefore the probability succeeds is at most:

$$q_{u^\star,v^\star} \leq (1-p)^{\lceil \frac{\omega(\log n)+\log_2(e)\cdot B}{p} \rceil} \leq 2^{\log_2(e)\cdot \overbrace{\log(1-p)}^{\leq -p} (\frac{\omega(\log n)+\log_2(e)\cdot B}{p}+1)} = n^{-\omega(1)}/2^B.$$

Finally, $q \leq q_{u^\star,v^\star} \cdot n(n-1) = n^{-\omega(1)}/2^B$. □

References

ACD+19. Ittai Abraham, T.-H. Hubert Chan, Danny Dolev, Kartik Nayak, Rafael Pass, Ling Ren, and Elaine Shi. Communication complexity of byzantine agreement, revisited. In Peter Robinson and Faith Ellen, editors, *38th ACM PODC*, pages 317–326. ACM, July / August 2019.

ACS22. Gilad Asharov, Ran Cohen, and Oren Shochat. Static vs. adaptive security in perfect MPC: A separation and the adaptive security of BGW. In Dana Dachman-Soled, editor, *ITC 2022*, volume 230 of *LIPIcs*, pages 15:1–15:16. Schloss Dagstuhl, July 2022.

AL17. Gilad Asharov and Yehuda Lindell. A full proof of the BGW protocol for perfectly secure multiparty computation. *Journal of Cryptology*, 30(1):58–151, January 2017.

ALM17. Adi Akavia, Rio LaVigne, and Tal Moran. Topology-hiding computation on all graphs. In Jonathan Katz and Hovav Shacham, editors, *CRYPTO 2017, Part I*, volume 10401 of *LNCS*, pages 447–467. Springer, Cham, August 2017.

AM17. Adi Akavia and Tal Moran. Topology-hiding computation beyond logarithmic diameter. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *EUROCRYPT 2017, Part III*, volume 10212 of *LNCS*, pages 609–637. Springer, Cham, April / May 2017.

BBC+19. Marshall Ball, Elette Boyle, Ran Cohen, Tal Malkin, and Tal Moran. Is information-theoretic topology-hiding computation possible? In Dennis Hofheinz and Alon Rosen, editors, *TCC 2019, Part I*, volume 11891 of *LNCS*, pages 502–530. Springer, Cham, December 2019.

BBC+20. Marshall Ball, Elette Boyle, Ran Cohen, Lisa Kohl, Tal Malkin, Pierre Meyer, and Tal Moran. Topology-hiding communication from minimal assumptions. In Rafael Pass and Krzysztof Pietrzak, editors, *TCC 2020, Part II*, volume 12551 of *LNCS*, pages 473–501. Springer, Cham, November 2020.

BBC24. D'or Banoun, Elette Boyle, and Ran Cohen. Information-theoretic topology-hiding broadcast: Wheels, stars, friendship, and beyond. In Divesh Aggarwal, editor, *ITC 2024*, volume 304 of *LIPIcs*, pages 1:1–1:13. Schloss Dagstuhl, August 2024.

BBCL23. Erica Blum, Elette Boyle, Ran Cohen, and Chen-Da Liu-Zhang. Communication lower bounds for cryptographic broadcast protocols. In *37th International Symposium on Distributed Computing, DISC 2023, October 10-12, 2023, L'Aquila, Italy*, volume 281 of *LIPIcs*, pages 10:1–10:19. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2023.

BBKM23. Marshall Ball, Alexander Bienstock, Lisa Kohl, and Pierre Meyer. Towards topology-hiding computation from oblivious transfer. In Guy N. Rothblum and Hoeteck Wee, editors, *TCC 2023, Part I*, volume 14369 of *LNCS*, pages 349–379. Springer, Cham, November / December 2023.

BCDH18. Elette Boyle, Ran Cohen, Deepesh Data, and Pavel Hubácek. Must the communication graph of MPC protocols be an expander? In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2018, Part III*, volume 10993 of *LNCS*, pages 243–272. Springer, Cham, August 2018.

BCG21. Elette Boyle, Ran Cohen, and Aarushi Goel. Breaking the $O(\sqrt{(}n))$-bit barrier: Byzantine agreement with polylog bits per party. In Avery Miller, Keren Censor-Hillel, and Janne H. Korhonen, editors, *40th ACM PODC*, pages 319–330. ACM, July 2021.

BCP15. Elette Boyle, Kai-Min Chung, and Rafael Pass. Large-scale secure computation: Multi-party computation for (parallel) RAM programs. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *CRYPTO 2015, Part II*, volume 9216 of *LNCS*, pages 742–762. Springer, Berlin, Heidelberg, August 2015.

BGT13. Elette Boyle, Shafi Goldwasser, and Stefano Tessaro. Communication locality in secure multi-party computation - how to run sublinear algorithms in a distributed setting. In Amit Sahai, editor, *TCC 2013*, volume 7785 of *LNCS*, pages 356–376. Springer, Berlin, Heidelberg, March 2013.

BGW88. Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract). In *20th ACM STOC*, pages 1–10. ACM Press, May 1988.

BLZL+25. Amey Bhangale, Chen-Da Liu-Zhang, Julian Loss, Kartik Nayak, and Sravya Yandamuri. Leader election with poly-logarithmic communication per party. In Yael Tauman Kalai and Seny F. Kamara, editors, *CRYPTO 2025, Part II*, volume 16001 of *LNCS*, pages 37–68. Springer, Cham, August 2025.

Can00. Ran Canetti. Security and composition of multiparty cryptographic protocols. *Journal of Cryptology*, 13(1):143–202, January 2000.

Can01. Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *42nd FOCS*, pages 136–145. IEEE Computer Society Press, October 2001.

CCD88. David Chaum, Claude Crépeau, and Ivan Damgård. Multiparty unconditionally secure protocols (extended abstract). In *20th ACM STOC*, pages 11–19. ACM Press, May 1988.

CCG+15. Nishanth Chandran, Wutichai Chongchitmate, Juan A. Garay, Shafi Goldwasser, Rafail Ostrovsky, and Vassilis Zikas. The hidden graph model: Communication locality and optimal resiliency with adaptive faults. In Tim Roughgarden, editor, *ITCS 2015*, pages 153–162. ACM, January 2015.

CCGZ19. Ran Cohen, Sandro Coretti, Juan A. Garay, and Vassilis Zikas. Probabilistic termination and composability of cryptographic protocols. *Journal of Cryptology*, 32(3):690–741, July 2019.

CCGZ21. Ran Cohen, Sandro Coretti, Juan A. Garay, and Vassilis Zikas. Round-preserving parallel composition of probabilistic-termination cryptographic protocols. *Journal of Cryptology*, 34(2), 2021.

CDD+99. Ronald Cramer, Ivan Damgård, Stefan Dziembowski, Martin Hirt, and Tal Rabin. Efficient multiparty computations secure against an adaptive adversary. In Jacques Stern, editor, *EUROCRYPT'99*, volume 1592 of *LNCS*, pages 311–326. Springer, Berlin, Heidelberg, May 1999.

CFGN96. Ran Canetti, Uriel Feige, Oded Goldreich, and Moni Naor. Adaptively secure multi-party computation. In *28th ACM STOC*, pages 639–648. ACM Press, May 1996.

CGM+24. Nishanth Chandran, Juan A. Garay, Ankit Kumar Misra, Rafail Ostrovsky, and Vassilis Zikas. Adaptive security, erasures, and network assumptions in communication-local MPC. In Elette Boyle and Mohammad Mahmoody, editors, *TCC 2024, Part IV*, volume 15367 of *LNCS*, pages 293–326. Springer, Cham, December 2024.

CGO12a. Nishanth Chandran, Juan Garay, and Rafail Ostrovsky. Almost-everywhere secure computation with edge corruptions. Cryptology ePrint Archive, Report 2012/221, 2012.

CGO12b. Nishanth Chandran, Juan A. Garay, and Rafail Ostrovsky. Edge fault tolerance on sparse networks. In Artur Czumaj, Kurt Mehlhorn, Andrew M. Pitts, and Roger Wattenhofer, editors, *ICALP 2012, Part II*, volume 7392 of *LNCS*, pages 452–463. Springer, Berlin, Heidelberg, July 2012.

CGZ23. Ran Cohen, Juan A. Garay, and Vassilis Zikas. Completeness theorems for adaptively secure broadcast. In Helena Handschuh and Anna Lysyanskaya, editors, *CRYPTO 2023, Part I*, volume 14081 of *LNCS*, pages 3–38. Springer, Cham, August 2023.

Cha88. David Chaum. The dining cryptographers problem: Unconditional sender and recipient untraceability. *Journal of Cryptology*, 1(1):65–75, January 1988.

CLOS02. Ran Canetti, Yehuda Lindell, Rafail Ostrovsky, and Amit Sahai. Universally composable two-party and multi-party secure computation. In *34th ACM STOC*, pages 494–503. ACM Press, May 2002.

DI05. Ivan Damgård and Yuval Ishai. Constant-round multiparty computation using a black-box pseudorandom generator. In Victor Shoup, editor, *CRYPTO 2005*, volume 3621 of *LNCS*, pages 378–394. Springer, Berlin, Heidelberg, August 2005.

DKM+17. Varsha Dani, Valerie King, Mahnush Movahedi, Jared Saia, and Mahdi Zamani. Secure multi-party computation in large networks. *Distributed Computing*, 30(3):193–229, 2017.

DN14. Ivan Damgård and Jesper Buus Nielsen. Adaptive versus static security in the UC model. In Sherman S. M. Chow, Joseph K. Liu, Lucas C. K. Hui, and Siu-Ming Yiu, editors, *ProvSec 2014*, volume 8782 of *LNCS*, pages 10–28. Springer, Cham, October 2014.

Dol82. Danny Dolev. The byzantine generals strike again. *Journal on Algorithms*, 3(1):14–30, 1982.

Fel88. Paul Feldman. *Optimal Algorithms for Byzantine Agreement*. PhD thesis, Stanford University, 1988. https://dspace.mit.edu/handle/1721.1/14368.

FGK24. Rex Fernando, Yuval Gelles, and Ilan Komargodski. Scalable distributed agreement from LWE: Byzantine agreement, broadcast, and leader election. In Venkatesan Guruswami, editor, *ITCS 2024*, volume 287, pages 46:1–46:23. LIPIcs, January / February 2024.

FLM86. Michael J. Fischer, Nancy A. Lynch, and Michael Merritt. Easy impossibility proofs for distributed consensus problems. *Distributed Computing*, 1(1):26–39, 1986.

GK24. Yuval Gelles and Ilan Komargodski. Optimal load-balanced scalable distributed agreement. In Bojan Mohar, Igor Shinkar, and Ryan O'Donnell, editors, *56th ACM STOC*, pages 411–422. ACM Press, June 2024.

GKKZ11. Juan A. Garay, Jonathan Katz, Ranjit Kumaresan, and Hong-Sheng Zhou. Adaptively secure broadcast, revisited. In Cyril Gavoille and Pierre Fraigniaud, editors, *30th ACM PODC*, pages 179–186. ACM, June 2011.

GMW87. Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or A completeness theorem for protocols with honest majority. In Alfred Aho, editor, *19th ACM STOC*, pages 218–229. ACM Press, May 1987.

Gol04. Oded Goldreich. *Foundations of Cryptography: Basic Applications*, volume 2. Cambridge University Press, Cambridge, UK, 2004.

HMTZ16. Martin Hirt, Ueli Maurer, Daniel Tschudi, and Vassilis Zikas. Network-hiding communication and applications to multi-party protocols. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part II*, volume 9815 of *LNCS*, pages 335–365. Springer, Berlin, Heidelberg, August 2016.

HZ10. Martin Hirt and Vassilis Zikas. Adaptively secure broadcast. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 466–485. Springer, Berlin, Heidelberg, May / June 2010.

KLST11. Valerie King, Steven Lonargan, Jared Saia, and Amitabh Trehan. Load balanced scalable byzantine agreement through quorum building, with full information. In *Proceedings of the 12th International Conference on Distributed Computing and Networking (ICDCN)*, pages 203–214, 2011.

KS10. Valerie King and Jared Saia. Breaking the $O(n^2)$ bit barrier: scalable byzantine agreement with an adaptive adversary. In Andréa W. Richa and Rachid Guerraoui, editors, *29th ACM PODC*, pages 420–429. ACM, July 2010.

KSSV06. Valerie King, Jared Saia, Vishal Sanwalani, and Erik Vee. Scalable leader election. In *17th SODA*, pages 990–999. ACM-SIAM, January 2006.

MOR15. Tal Moran, Ilan Orlov, and Silas Richelson. Topology-hiding computation. In Yevgeniy Dodis and Jesper Buus Nielsen, editors, *TCC 2015, Part I*, volume 9014 of *LNCS*, pages 159–181. Springer, Berlin, Heidelberg, March 2015.

Yao82. Andrew Chi-Chih Yao. Protocols for secure computations (extended abstract). In *23rd FOCS*, pages 160–164. IEEE Computer Society Press, November 1982.