Information-Theoretic Topology-Hiding Broadcast: Wheels, Stars, Friendship, and Beyond[*]

D'or Banoun[†] Elette Boyle[‡] Ran Cohen[§]

August 9, 2024

Abstract

Topology-hiding broadcast (THB) enables parties communicating over an incomplete network to broadcast messages while hiding the network topology from within a given class of graphs. Although broadcast is a privacy-free task, it is known that THB for certain graph classes necessitates computational assumptions, even against "honest but curious" adversaries, and even given a single corrupted party. Recent works have tried to understand when THB can be obtained with *information-theoretic* (IT) security (without cryptography or setup assumptions) as a function of properties of the corresponding graph class.

We revisit this question through a case study of the class of *wheel* graphs and their subgraphs. The n^{th} wheel graph is established by connecting n nodes who form a cycle with another "center" node, thus providing a natural extension that captures and enriches previously studied graph classes in the setting of IT-THB.

We present a series of new findings in this line. We fully characterize feasibility of IT-THB for any class of subgraphs of the wheel, each possessing an embedded *star* (i.e., a well-defined center connected to all other nodes). Our characterization provides evidence that IT-THB feasibility may correlate with a more fine-grained degree structure—as opposed to pure connectivity—of the corresponding graphs. We provide positive results achieving *perfect* IT-THB for new graph classes, including ones where the number of nodes is unknown. Further, we provide the first feasibility of IT-THB on non-degenerate graph-classes with $t > 1$ corruptions, for the class of *friendship* graphs (Erdös, Rényi, Sós'66).

[*] A preliminary version of this work appeared in *ITC 2024*.

[†] Reichman University. E-mail: dor.banoun@post.runi.ac.il. Supported in part by AFOSR Award FA9550-21-1-0046.

[‡] Reichman University and NTT Research. E-mail: elette.boyle@runi.ac.il. Supported in part by AFOSR Award FA9550-21-1-0046 and ERC Project HSS (852952).

[§] Reichman University. E-mail: cohenran@runi.ac.il. Supported in part by NSF grant no. 2055568, by ISF grant 1834/23, and by the Algorand Centres of Excellence programme managed by Algorand Foundation. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of Algorand Foundation.

Contents

1 Introduction

Topology-hiding protocols over an incomplete communication network guarantee that colluding parties do not learn additional information about the topology of the network graph (from within a given class of graphs), beyond their own neighbor-set [MOR15]. Such protocols may be of interest in settings where the communication structure itself is sensitive information, such as in social networks, or peer-to-peer networks based on geographical position. Perhaps the most fundamental goal is that of achieving topology-hiding *broadcast* (THB), where a designated sender wishes to convey an input to all participating parties.

Although broadcast is a privacy-free task, THB turned out to be a challenging goal on its own. It was recently shown that THB for certain graph classes necessitates computational assumptions, even in the "honest but curious" *semi-honest* setting (when corrupted parties follow the protocol honestly but try to learn more information from their joint view), and even given a *single* corrupted party [BBMM18, BBC+19]. This lies in stark contrast to the topology-revealing case, in which broadcast is trivially achievable in the semi-honest setting.

Obtaining topology hiding based on computational assumptions has been the subject of a fruitful collection of works, leading to various THB, and in turn, general topology-hiding secure multiparty computation (THC) protocols [MOR15, HMTZ16, AM17, ALM17, LZM+18, BBMM18, LZM+20, Li22, BBKM23]. It is known by now how to construct THB protocols for the class of all graphs (of polynomial size) that are secure against *any* subset of semi-honest corruptions under standard number-theoretic cryptographic hardness assumptions such as DDH, QR, and LWE,[1] or from unstructured assumptions such as constant-round constant-rate oblivious transfer [BBKM23].

Motivated by an analogous question within secure multi-party computation, the work of [BBC+19] asked whether existence of an honest majority can enable *information-theoretically* secure THB protocols in certain settings, without relying on cryptographic assumptions and withstanding computationally unbounded adversaries. We refer to this as IT-THB. The work of [BBC+19] ruled out 1-secure IT-THB on a path with four nodes (which is 1-connected) but devised a perfect 1-secure information-theoretic THC on cycles of known length (which are 2-connected); see Figure 1. Given these initial evidence, they conjectured that feasibility of IT-THB may depend on the *connectivity*[2] of the graphs within the class: namely, that $(t+1)$-connectivity is sufficient and/or necessary for t-secure IT-THB.

The special case of $t = 1$ was further investigated by [BBC+20], who proved that the conjecture holds in this case for the stronger notion of THC. They showed that information-theoretic THC with security against a single semi-honest corruption is possible if and only if the connectivity of every graph in the class is at least 2. However, they additionally showed that the conjecture does *not* hold for THB, by constructing a perfectly secure THB against a single corruption for the butterfly graph class (Figure 1), where each graph is only 1-connected.

The results of [BBC+19, BBC+20] open a rich domain of questions. As [BBC+20] showed, high connectivity is *not* the "right" criterion for feasibility of THB (in contrast to THC), and alternative graph-properties may serve as candidate conjectures. Therefore, our first question is:

Given a graph-class, which graph properties characterize feasibility of 1*-secure IT-THB?*

[1] DDH stands for the decisional Diffie-Hellman assumption, QR for the quadratic residuosity assumption, and LWE for the learning with errors assumption.

[2] We consider node-connectivity; that is, a graph is k-connected if and only if every pair of nodes is connected by k *vertex-disjoint* paths.

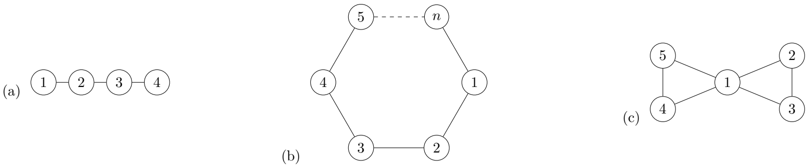


Figure 1: (a) Class $\mathcal{G}_{\textsf{4-path}}$, of all isomorphisms of 4 nodes on a path; 1-secure THB over $\mathcal{G}_{\textsf{4-path}}$ implies key agreement. (b) Class $\mathcal{G}_{\textsf{cycle}}(n)$ of all isomorphisms of n nodes on a cycle; admits 1-secure IT-THB. (c) Class $\mathcal{G}_{\textsf{butterfly}}$ of all isomorphisms of 5 nodes on a butterfly graph (two triangles with a common node); contains 1-connected graphs yet admits 1-secure IT-THB.

Zooming into [BBC+20], their general positive result, of 1-IT-THB over 2-connected graphs, has a nonzero (yet exponentially small) error probability. This means that non-trivial THB with *perfect* security is only known for cycles [BBC+19] and for the butterfly graph [BBC+20]. Is it possible that for graphs with $n > 5$ nodes the source of perfect 1-THB is the highly symmetric structure of cycles? Do other graph classes inherently require a positive error?

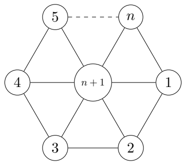
Are there additional graph-classes that support perfectly secure THB?

Finally, all feasibility results for IT-THB are secure against a single corruption. Indeed, 2-secure THB on a 4-node rectangle, possibly with a missing edge, requires oblivious transfer [BBMM18], and a 2-secure THB on a cycle with 7 nodes (or more) requires key agreement [BBC+19]. The statistically secure THB protocols for 2-connected graphs from [BBC+20] completely break if there are two corruptions, and in the butterfly class two corruptions trivialize the problem, as there is no information to hide. One may wonder if IT-THB simply cannot withstand multiple corruptions that provide several points of view about the graph topology, except for degenerate cases where the topology is already revealed by the corrupted parties' neighbor-sets. This leads to our third question:

Are there graph-classes that support IT-THB with more than a single corruption?

1.1 Our Contributions

In this work, we conduct an investigation of these questions through a case study of the class of *wheel graphs* and their subgraphs. The n^{th} wheel graph W_n is established by connecting a single node (the "center") to n nodes who form a cycle, as depicted below. The wheel graph-class $\mathcal{G}_{\textsf{wheel}}(n)$ consists of all isomorphisms of the wheel graph, i.e., all assignments of the labels $\{1, \ldots, n+1\}$ to the nodes of the wheel graph W_n.



Wheel graphs and their subgraphs form a natural extension that captures and enriches previously studied graph classes in the setting of IT-THB: for example, paths, cycles, triangles, and butterfly graphs. Interestingly, although $\mathcal{G}_{\textsf{wheel}}(n)$ has increased connectivity over the n-cycles, the corresponding state-of-the-art THB protocols for $\mathcal{G}_{\textsf{wheel}}(n)$ are slightly worse. Note that the cycle protocol cannot simply be run directly, as parties on the perimeter of the graph do not know—in fact, must not know—which neighbor is the center node.

Several challenges arise when hiding the topology of $\mathcal{G}_{\mathsf{wheel}}(n)$. First, consider a node v on the perimeter; such a node has three neighbors, one of which is the center. To hide the identity of the center node, either the protocol does not utilize the power of the center, or each of the non-center neighbors must emulate the behavior of the center toward v, and further, v must emulate the center toward all its neighbors. Second, consider the center node; this node is connected to all other parties but must not learn how the parties on the perimeter are connected among themselves. Further, an adversary that corrupts two parties on the perimeter without a common neighbor must not learn their relative distance on the perimeter. Note that an adversary that corrupts $n-2$ nodes in the $(n+1)$-nodes wheel graph knows the entire topology from the corrupted nodes' neighbor-sets; however, for $t \leq n-3$ corruptions not all is revealed (i.e., when there are 4 honest parties).[3]

Characterization of wheels and subgraphs with an embedded star. Our first result shows that perfectly secure THB is possible against a single semi-honest corruption on the class of wheel graphs $\mathcal{G}_{\mathsf{wheel}}(n)$, as well as on certain classes of its subgraphs. Concretely, given any family of subgraphs of the wheel with $n+1$ nodes, with an embedded star in each graph (i.e., where the center is fully connected and has degree n), we show that $\mathsf{IT\text{-}THB}$ with one corruption is possible if either the *minimal degree* of non-center nodes in the family is greater than 1, or if it is 1 but so is the *maximal degree*. Surprisingly, we show that this characterization is tight for any such subclasses that are closed under isomorphism (i.e., for each graph topology in the class, all relabelings of this graph are also contained in the class); that is, if the maximal degree is greater than 1 but the minimal degree is 1, then THB on this class implies key agreement.

This would suggest that feasibility of $\mathsf{IT\text{-}THB}$ may correlate with a more fine-grained degree structure, as opposed to connectivity, of graphs.

More concretely, we begin by defining *admissible subgraphs* as subgraphs of the wheel graph W_n in which the degree of the center is n and the degree of every other node is either 2 or 3. The *butterfly graph* is an example for an admissible subgraph for $n = 4$, as well as the $(2n+1)$-node *friendship graph* F_n,[4] see Figure 2.

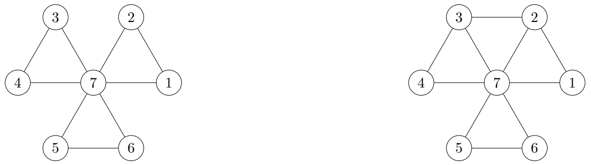


Figure 2: Examples of admissible subgraphs of $\mathcal{G}_{\mathsf{wheel}}(6)$. On the left is a friendship graph in which every non-center node has degree 2, and on the right is a subgraph where every non-center node has degree 2 or 3.

When considering graphs with an embedded star, i.e., with a fully connected center, *non-admissible graphs* are those who contain a non-center node of degree 1. The extreme example is the *star graph* in which the center node is connected to n nodes, and no other edges exist, see Figure 3.

[3] When considering arbitrary admissible graphs with $n+1$ nodes (as defined below), there is more information to hide; therefore, an adversary that corrupts n nodes knows the entire topology but for $t \leq n-2$ not all is revealed (i.e., when there are 3 honest parties) .

[4] The friendship graph F_n, introduced in [ERS66], is a planar, undirected graph with $2n+1$ nodes and $3n$ edges. F_n can be constructed by joining n triangles with a common node.

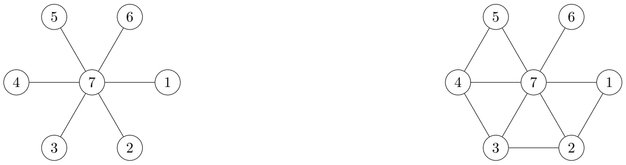


Figure 3: Example of non-admissible subgraphs of $\mathcal{G}_{\mathsf{wheel}}(6)$. On the left is the star graph with 7 nodes. On the right is a subgraph with a single node of degree 1.

Our characterization nearly shows that IT-THB is possible for a given graph-class with a fully connected center if and only if it consists only of admissible subgraphs. The single exception is the graph class $\mathcal{G}_{\mathsf{star}}(n)$ that only contains star graphs, which are not admissible; this class is degenerate (trivially hides the topology) since any node can identify the center and derive the whole topology.

Theorem 1.1 (IT-THB for admissible graphs with fixed size, informal)**.** *Let $n \in \mathbb{N}$ with $n \geq 4$, and let $\mathcal{G} \subseteq \mathcal{G}_{\mathsf{wheel}}(n)$ be a graph-class in which every graph has $n+1$ nodes and the center has degree n.*

Then, if either $\mathcal{G} = \mathcal{G}_{\mathsf{star}}(n)$ or if $\mathcal{G}$ consists of admissible graphs, there exists perfectly secure IT-THB against a single semi-honest corruption over $\mathcal{G}$. Otherwise, THB over $\mathcal{G}$ secure against a single semi-honest corruption exists if and only if key agreement exists.

Theorem 1.1 demonstrates another interesting phenomena: a nontrivial example of a graph-class in which $\mathcal{G}$ is the union of two sub-classes $\mathcal{G}_1$ and $\mathcal{G}_2$, such that each sub-class admits an IT-THB, yet the there is no IT-THB for $\mathcal{G}$. Specifically, while $\mathcal{G}_{\mathsf{wheel}}(n)$ and $\mathcal{G}_{\mathsf{star}}(n)$ each individually admits 1-IT-THB, any 1-THB protocol on $\mathcal{G}_{\mathsf{wheel}}(n) \cup \mathcal{G}_{\mathsf{star}}(n)$ requires key agreement.

Generalizing to variable-size subgraphs. We proceed to analyze subgraphs of $\mathcal{G}_{\mathsf{wheel}}(n)$ that are generated by removing some of the nodes. Note that when disconnecting the center node, the resulting subgraph is either a cycle with n nodes $\mathcal{G}_{\mathsf{cycle}}(n)$, which supports 1-secure perfect THB, or a path with up to n nodes that necessitates key agreement. Therefore, we focus on keeping the center and disconnecting nodes from the perimeter. An interesting observation is that when disconnecting k neighboring nodes from the perimeter, the result is an admissible subgraph of the wheel with $n+1-k$ nodes with one edge removed from the perimeter. Similarly, disconnecting arbitrary k nodes yields a subgraph of the wheel with $n+1-k$ nodes with m edges removed from the perimeter, where m is the number of sets of neighboring nodes that are removed.

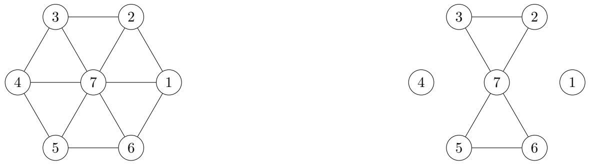


Figure 4: On the left is a wheel graph. On the right is the resulting graph when disconnecting nodes 1 and 4 by removing their corresponding edges. The result is the butterfly graph F_2 and two isolated nodes.

A more interesting question is thus to characterize families of such subgraphs whose number of nodes is not a priori known. We remark that topology hiding on graphs of unknown size can be surprisingly complex: For example, THB with an additional sender-anonymity guarantee for the simple class of 2-paths and 3-paths implies *infinitely often oblivious transfer* [BBC+20, Thm 5.4].

We utilize a useful property of the protocol used for proving Theorem 1.1 (discussed further in Section 1.2) that effectively hides the number of nodes from non-center parties. We show that the protocol can be applied also to the current setting to obtain perfect IT-THB.

Theorem 1.2 (IT-THB for admissible graphs with varying size, informal)**.** *Let $n \in \mathbb{N}$ and let $\mathcal{G}$ be a graph-class such that every $(V, E) \in \mathcal{G}$ is a subgraph of the wheel graph W_n, and it holds that $4 \leq |V| \leq n+1$ and that there is a center node with degree $|V|-1$. Then,*

- *if the maximal degree of non-center nodes is 1, i.e., $\mathcal{G}$ consists only of stars (possibly of different size), or*
- *if the minimal degree of non-center nodes is 2 or 3, i.e., $\mathcal{G}$ consists only of admissible graphs, or*
- *if $\mathcal{G}$ consists both of stars and admissible graphs but they are of different sizes,*

there exists perfectly secure IT-THB against a single semi-honest corruption over $\mathcal{G}$. Otherwise, THB over $\mathcal{G}$ secure against a single semi-honest corruption exists if and only if key agreement exists.

We note that Theorem 1.2 subsumes Theorem 1.1; therefore, in the technical sections we directly prove Theorem 1.2.

Tolerating many corruptions: the case of friendship graphs. The feasibility results thus far were limited to a single corruption. The reason lies in the structure of the protocol, which enables two colluding parties with two common neighbors to learn which of them is the center; see Section 1.2 for an illustration. Therefore, it still remains open whether IT-THB tolerating $t > 1$ corruption is possible, aside from degenerate cases in which the topology is fully determined from neighbor-sets of any t nodes.

We proceed to analyze an interesting class of subgraphs of a wheel graph with varying size, which consists of *friendship graphs*. Recall that for $n \geq 1$, the friendship graph F_n is a $(2n+1)$-nodes graph constructed by joining n triangles with a common node. They were named after the friendship theorem [ERS66], which states that if in a finite set of people every pair has one common friend, then there exists one person who is friend with everyone. We consider a class consisting of friendship graphs of different sizes. Note that the connectivity of each of those graphs is 1, and by their structure every two nodes can only have one common neighbor, so the attack discussed above no longer applies. We prove that indeed perfect IT-THB tolerating *any* number of corruptions can be achieved on this class. For an integer k, consider the graph class $\mathcal{G}_{\mathsf{friendship}}(k)$ containing all isomorphisms of the friendship graph F_k.

Theorem 1.3 (t-IT-THB over friendship graphs, informal)**.** *Let $n \in \mathbb{N}$ with $n \geq 2$, let $t < 2n+1$, and consider a graph-class $\mathcal{G} \subseteq \bigcup_{k=2}^{n} \mathcal{G}_{\mathsf{friendship}}(k)$. There exists a perfectly secure THB protocol against t semi-honest corruptions over $\mathcal{G}$.*

We remark that Theorem 1.3 presents the first feasibility of information-theoretic THB on non-degenerate graph-classes with $t > 1$ corruptions.

1.2 Technical Overview

We move on to describing some of our techniques. We begin by explaining in Section 1.2.1 the high-level ideas of the protocols used for our positive result. Next, in Section 1.2.2, we describe our usage of the *phantom-jump* technique from [BBC+20] for our negative result.

1.2.1 Feasibility Results: The "Oblivious Centralized Coordination" Technique

Our protocols are inspired by the THB protocol for the butterfly graph from [BBC+20]. We extend it in several aspects to support more involved graph classes that contain an embedded star, i.e., a well-defined center connected to all other nodes. In the overview below, as well as in the technical sections, we begin by describing the simpler case of friendship graphs, and then proceed to the wheel graph, and to arbitrary admissible graphs.

Starting point: the butterfly graph. Recall that the butterfly graph (Figure 1) is in fact the friendship graph F_2: a 5-node graph consisting of two triangles connected by a common center node. The high-level idea is to use the center node for coordinating the protocol. The protocol runs multiple instances of *reliable message transmission* (RMT), one for every potential receiver. In each RMT instance, the sender P_S sends its message to all its neighbors in the first step. Note that each party knows whether it is a neighbor of P_S, so it knows whether it should receive a message or not in the first round. At that point it is guaranteed that the center node holds the message and so can deliver it to the receiver (in case the receiver is not the center).

This, of course, will reveal to the receiver who is the center node. Therefore, the center must do so in an *oblivious* way, without exposing itself. In the butterfly graph, if the receiver P_R is not the center it has one more neighbor other than the center. The approach taken in [BBC+20] is to first secret share the message m with the additional neighbor, and later have both neighbors of the receiver deliver one share (thus hiding from the receiver who is the center). However, the center does not know who is the additional neighbor of the receiver. Therefore, the center node prepares 2-out-of-2 shares of the message m for each potential neighbor, i.e., each non-receiver party.

To help the center hide its identity, each other party assists by acting as the center and preparing 2-out-of-2 shares of zero (so called, *blinding terms for addition* in Section 3) for each of its non-receiver neighbors (a non-center party has either one or two non-receiver neighbors). Next, the receiver receives four values from each of its neighbors (recall that in the butterfly graph there are four nodes other than the receiver, see Figure 1), such that the center sends the sum of the share m for each party with the share of zero it received from that party, and the second neighbor sends the sum of the share received from its non-receiver neighbor with the share of zero sent to this neighbor, along with three random values (one for each other party). The receiver can then select the correct pair which corresponds to its true neighbors. Thus, P_R can reconstruct m without knowing which of its neighbors is the center.

This approach is secure as long as the receiver is not the center. However, if P_R is the center, it may learn the neighbor-set of other nodes (e.g., by inspecting which pairs of values sum up to 0). This is solved by adding *suitable offset* values, which are multiplied by *blinding terms for multiplication*, and only come into play if P_R is the center. Specifically, if P_R is *not* the center, then P_R will send the same offset to both its neighbors (this will ensure that the offset will be canceled out). If P_R is the center, then P_R will send a *different* offset to each neighbor (this requires working over a larger field, e.g., $\mathbb{F}_4$, to support a different value per party); in this case, the pairs of values seen by P_R will induce a linear system of two equations with two variables, and the different offsets will guarantee that the system has full rank and always has a solution. This, in turn, will prove that the center cannot identify which pairs of parties are connected.

The friendship graph. As discussed above, we view the butterfly graph as two triangles connected in a joint node; that is, as the friendship graph F_2. In Section 3.1, we prove that the 1-THB protocol for the butterfly graph-class $\mathcal{G}_{\mathsf{butterfly}}$ (consisting of all isomorphisms of 5 nodes to F_2) extends in natural way to 1-THB for the class of friendship graph $\mathcal{G}_{\mathsf{friendship}}(n)$, for $n \geq 2$, consisting of all isomorphisms of $2n + 1$ nodes to F_n.[5] Namely, the receiver now receives a vector of $2n$ values from each of its neighbors, and those values are uniformly distributed conditioned on the corresponding values of its neighbors that sum up to the message. Further, recall that in case the receiver is the center, it must provide a different offset to each of its neighbors; hence, the underlying field $\mathbb{F}_q$ must grow and satisfy $q \geq 2n$.

Friendship of variable size. A second observation is that for non-center parties, the protocol behaves in a "local" manner, in the sense that the neighbors of a non-center node are neighbors on their own. When P_R is not the center, this enables the receiver's neighbors to jointly construct the shares of the message in a coordinated (yet oblivious) way. Only the center's actions truly depend on the actual number of parties, while non-center parties only need to know an upper bound on the number of parties.

In Section 3.2, we prove that this locality property makes the protocol suitable for a variable number of nodes (i.e., a variable number of triangles). Non-center nodes proceed as if the graph has n triangles (where n is an upper bound), and the center node emulates missing nodes in its head. Formally, we consider the graph $F_{k,n}$ as an *augmented friendship graph* of $2n + 1$ nodes, where $2k + 1$ nodes form a connected component which is the friendship graph F_k, and all other $2n + 1 - (2k + 1) = 2(n - k)$ nodes are singletons (isolated parties). Each isolated party simply outputs 0 in this protocol (unless it is the sender, in which case it outputs its input), and the *agreement* and *validity* properties are only required for the connected component of the sender.

Friendship with many corruptions. Another interesting observation, is that locality enables tolerating an arbitrary number of $t < 2n+1$ corruptions, *without* any adjustments to the protocol. We prove this in Section 3.3. Intuitively, to see why, we distinguish between an honest center and a corrupt center.

In case the center is honest, then once there is more than a single corruption, the adversary can immediately identify who the center is. This is not considered a violation of privacy, since this can be deduced just by observing the common neighbor of the corrupted parties, and *without* observing any protocol messages. When focusing on each triangle now, if both non-center nodes are corrupted there is nothing to hide within the triangle, whereas if none of the non-center nodes is corrupted the adversary learns nothing new from the protocol. The case where there is a single corrupted non-center in the triangle reduces to the single corruption case from before.

In case the center is corrupted, and there is another non-center corrupted party, then all the information in its triangle is already known, regardless of whether the second non-center party is honest or not. Further, consider the set of honest parties that have an honest neighbor, then the center together with all other corrupt parties do not learn the connectivity of this set.

[5] Note that for $n = 1$ a friendship graph is just a triangle, and there is no well-defined center.

We note that despite the technical simplicity of this result, it bares a more significant conceptual contribution, as it provides the first feasibility of IT-THB with more than one corruption beyond trivial graph classes.

Beyond friendship: the wheel graph. We proceed to extend the *oblivious centralized coordination* technique to more involved graph classes that admit an embedded star. As before, we begin by considering a *single* corruption. One can view the $(2n+1)$-nodes friendship graph F_n as a subgraph of the wheel W_{2n} in which every non-center node has degree 2. The wheel graph presents the other extreme in some sense, as every non-center node has degree 3.

A first attempt to extend the protocol to this new regime, is to use 3-out-of-3 secret sharing instead of 2-out-of-2. Stated differently, before, in F_n, if P_R is not the center it receives a *vector* of $2n$ values from each of its two neighbors such that the matching pair of values sum up to the message and all other values are independently and uniformly distributed. When considering the $(n+1)$-nodes wheel graph W_n, if P_R is not the center then it has 3 neighbors, and it receives a *matrix* of $n \times n$ values from each of its neighbors such that the corresponding entries in these matrices[6] sum up to the message and all other values are independently and uniformly distributed.

However, as opposed to the friendship regime, once a non-center node has degree 3 the protocol loses its locality property, as now not all neighbors of the receiver are neighbors on their own, and so the matrices are not "synchronized" like the vectors in the previous case. Indeed, if done without care, this approach leads to an attack. The reason is that the preparation of entry (v, w) for the matrix of party P_u is done as follows: if v and w are not neighbors of u sample a random value; if only one is a neighbor use the value that this party sent before (to ensure it will cancel out); and if both parties are neighbors of u then take the sum of their values. Therefore, the receiver can identify repeating entries in a matrix to deduce pairs of neighboring parties, as illustrated in Figure 5.

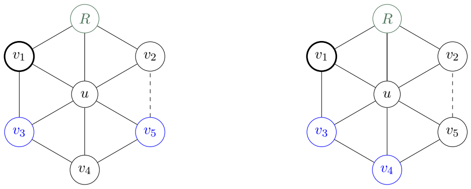


Figure 5: Illustration of an attack on a naïve protocol for $\mathcal{G}_{\mathsf{wheel}}(n)$. The receiver P_R has three neighbors: v_1, v_2, and the center u. Say that v_1 has another neighbor v_3, which has a third neighbor v_4, which has a third neighbor v_5. Then, P_R receives a matrix from v_1; however, since v_3 sends a single value to v_1, the entry (v_3, v_4) will be the same as the entry (v_3, v_5). This means that both v_4 and v_5 are *not* neighbors of v_1.

Our solution to this issue is to have each pair of neighbors (none of which is P_R) generate a vector of n *correlated values*, as opposed to a single value. This is done by having each party sample a vector of random values and send it to each of its non-receiver neighbors. In fact, those correlated values make the *blinding terms* and the *suitable-offset terms* redundant, so these values are no longer used in this protocol. In Section 4.1, we prove that the resulting protocol is secure for the class of wheel graphs.

[6] That is, for neighbors u, v, w take entry (u, v) from the matrix of w, entry (v, w) from the matrix of u, and entry (w, u) from the matrix of v. In the protocol, we ensure the matrices are symmetric, i.e., $\mathbf{M}[u, v] = \mathbf{M}[v, u]$.

Admissible graphs. Having established 1-IT-THB for the case when non-center nodes have degree 2 (friendship graphs) and the case where they have degree 3 (wheel graphs), we proceed to combine the ideas together and support any admissible graph. Intuitively, since the protocols share a similar structure, one can hope to execute both options concurrently. That is, the parties run two independent executions: one for the case where P_R has two neighbors, and one for the case where P_R has three neighbors. This, however, is vulnerable to an attack, since when a receiver has three neighbors it can find correlations in the messages it receives for the degree-2 execution and identify who the center is, as illustrated in Figure 6.

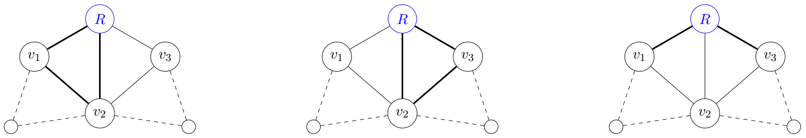


Figure 6: Illustration of an attack on a non-careful protocol for admissible graphs. Consider a non-center receiver P_R with neighbors v_1, v_2, and v_3; assume that v_2 is the center. Further, consider running the friendship protocol over this graph. The left diagram, illustrates the view P_R obtains for the triangle with v_1 and v_2: here P_R will obtain the message m. The middle diagram, illustrates the view P_R obtains for the triangle with v_2 and v_3: again, P_R will obtain the message m. The right diagram, illustrates the view P_R obtains for the triangle with v_1 and v_3: here, there is no direct edge between v_1 and v_3; hence, P_R will *not* obtain the message m. Therefore, P_R can identify that v_2 is the center.

The main idea in overcoming this attack, is that although we need to run two executions in order to hide the degree of the receiver (when it is not the center), we only need one execution to deliver the message to the receiver, and the second does not need to convey any information. Further, the receiver already knows its degree, so it knows which execution is the "right" one, and can sabotage the "redundant" one. Specifically:

- In case the receiver's degree is 3, in the degree-2 execution it will send a *different* offset for each neighbor (and the degree-3 execution will be executed correctly).
- In case the receiver's degree is 2, in the degree-2 execution it will correctly send the *same* offset to its neighbors (and the degree-3 execution will not leak any information because the receiver does not have three neighbors).

In Section 4.2, we prove that the resulting protocol is secure against one corruption for graph-classes consisting of admissible graphs.

Many corruptions. The protocol described above establishes feasibility of 1-IT-THB for any graph-class consisting of admissible graphs (even of variable size). This feasibility is tight for a single corruption, as stated in Theorem 1.2 (proven in Section 5). It is tempting though to extend the resiliency of the protocol, similarly to the class of friendship graphs that support any number of corruptions. It turns out that the non-local nature of non-friendship, admissible graphs enables an attack on the protocol when the adversary controls two nodes.

We illustrate the attack in Figure 7. Consider a pair of corrupted parties ② and ④ , and assume that none of them is the center. Further, assume that each has degree 3, and that they have two common neighbors, denoted ③ and ⑥ . Clearly, by the structure of the graph, ② and ④ together can deduce that either ③ is the center, or ⑥ is the center.

However, when running in this setting the 1-secure protocol described above, the colluding parties may learn correlations that will expose which of their common neighbors is the center. Specifically, recall that when ③ is the receiver, it sends to its neighbors ② and ④ the suitable-offset values. In case ③ is the center, the offset value for ② is the same as the one for ④ , whereas in case ③ is *not* the center these are different values.

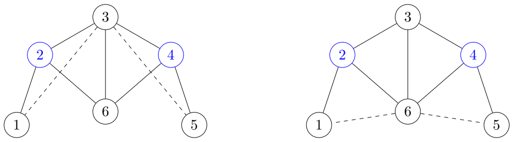


Figure 7: Attack on non-friendship admissible graphs with two corruptions.

We emphasize that in friendship graphs every non-center node has degree two; hence, the scenario from Figure 7 cannot occur. We leave it as an open question to find a protocol that is resilient to $t > 1$ corruptions for non-friendship admissible graphs.

1.2.2 Impossibility Results: The "Phantom Jump" Technique

The phantom-jump technique, introduced in [BBC$^+$20], was used to show that key agreement is necessary for 1-secure THB over the class $\mathcal{G}_{\mathsf{triangle}}$ consisting of a triangle, with possibly one of its edges missing (see Figure 8). In this class, if a party has two neighbors it does not know whether its neighbors are directly connected or not, but a party with one neighbor knows the entire topology.

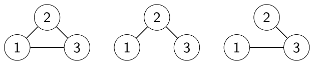


Figure 8: The class $\mathcal{G}_{\mathsf{triangle}}$ from [BBC$^+$20], consisting of a triangle, with possibly one of its edges missing.

In Section 5.1.1 we prove the lower bound of Theorem 1.1 (namely that 1-THB on the union of an admissible graph-class of size $n+1$ with $\mathcal{G}_{\mathsf{star}}(n)$ necessitates key agreement) by a direct reduction to the impossibility in [BBC$^+$20]. Below we explain in a more explicit manner how the phantom-jump technique from [BBC$^+$20] is used in this argument. We illustrate this for $\mathcal{G} = \mathcal{G}_{\mathsf{wheel}}(4) \cup \mathcal{G}_{\mathsf{star}}(4)$ where both graphs consist of 5 nodes.

The high-level idea, going back to [BBC$^+$19], is to construct a key-agreement protocol from a 1-secure THB protocol π for $\mathcal{G}$. Recall the desired key-agreement protocol is run between two parties, Alice and Bob, and concludes with the parties outputting a bit $b \in \{0,1\}$, such that a channel eavesdropper listening to communications cannot predict the value of b with non-negligible advantage. To construct a key-agreement protocol from π, Alice begins by choosing two long random strings m_1 and m_2 and sending them to Bob in the clear. Next, Alice and Bob continue in phases as follows:

- In each phase Alice and Bob locally toss coins A and B, respectively.
- They proceed to run two executions of π in which Alice always emulates ① and Bob always emulates ②. In addition, if $A = 0$ then Alice emulates ③ , ④ , and ⑤ as neighbors of ①, who acts as the center of the star, and ③ broadcasting m_1 in the first run; otherwise she emulates

③ , ④ , and ⑤ as neighbors of ①, who acts as the center of the star, and ③ broadcasting m_2 in the second run. Similarly, if $B = 1$ then Bob emulates ③ , ④ , and ⑤ as neighbors of ②, who acts as the center of the star, and ③ broadcasting m_1 in the first run; otherwise he emulates ③ , ④ , and ⑤ as neighbors of ②, who acts as the center of the star, and ③ broadcasting m_2 in the second run. See Figure 9 for an illustration.

- If parties ① and ② output m_1 in the first run and m_2 in the second, Alice and Bob output their bits A and B, respectively; otherwise, they execute another phase.

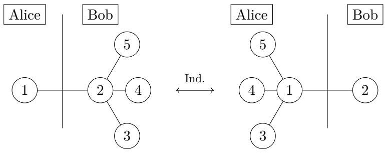


Figure 9: Using wheels and stars to construct a key-agreement protocol.

Clearly, if $A = B$ in some iteration then Alice and Bob will output the same coin, and by the assumed security of π, the eavesdropper Eve will not be able to learn who emulated ③ , ④ , and ⑤ in the first run and who in the second. If $A \neq B$, then in at least one of the runs nobody emulates the broadcaster ③ , so with overwhelming probability Alice and Bob will detect this case and execute another iteration.

In more detail, when $A = B$ the view of Eve consists of the communication between ① and ② , as depicted in Figure 9. By THB security, when ② acts as the center it cannot distinguish between the star and the wheel; in particular, the distribution of the messages on the channel between ① and ② is indistinguishable in both cases. Again, by THB security, when ① is not the center of the wheel it cannot know which of its neighbors is the center, so it cannot distinguish between the center being ② or ③ ; in particular, the distribution of the messages on the channel between ① and ② is indistinguishable in both cases. Similarly, when ② is not the center of the wheel, it cannot distinguish between the center being ① or ③ ; in particular, the distribution of the messages on the channel between ① and ② is indistinguishable in both cases. Finally, when ① acts as the center it cannot distinguish between the star and the wheel; in particular, the distribution of the messages on the channel between ① and ② is indistinguishable in both cases. By a simple hybrid argument it follows that the messages between ① and ② are indistinguishable when communicating in a star topology when ① is the center and when ② is the center, and it follows that the distinguishing advantage of Eve is negligible. See Figure 10 for an illustration of the hybrid argument.

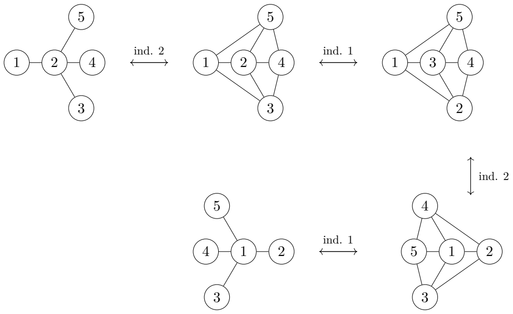


Figure 10: Hybrid steps in the phantom jump over wheels and stars.

Organization of the paper. In Section 2 we present the model and definitions. In Section 3 we describe our protocol for friendship graphs: initially for fixed-size graphs, next for variable-size graphs, and finally, for tolerating many corruptions. In Section 4 we describe our protocol for wheel graphs and the generalization to arbitrary admissible graphs. Finally, in Section 5 we present lower bounds and our characterization.

2 Preliminaries

Notations. For $n \in \mathbb{N}$ let $[n] = \{1, \ldots, n\}$. Looking ahead, we denote by n (or a function of n, such as $n+1$ or $2n+1$) an upper bound on the number of participating parties and by t an upper bound on the number of corrupted parties. The security parameter is denoted by κ. We denote variables in italic single-letter, e.g., x, vectors in lower-case bold-face, e.g., $\boldsymbol{v}$, the i^{th} location in the vector as $\boldsymbol{v}[i]$, matrices in upper-case bold-face, e.g., $\mathbf{M}$, and the $(i,j)^{\text{th}}$ location in the matrix as $\mathbf{M}[i,j]$, and Turing machines in sans-serif, e.g., P. We denote by $\mathbb{F}_q$ the finite field with q elements, by $\mathbb{F}_q^n$ the space of n-tuples over $\mathbb{F}_q$, and by $\mathbb{F}_q^{n\times n}$ the space of matrices of size $n \times n$ over $\mathbb{F}_q$.

Graph notations and properties. A graph $G = (V, E)$ is a set V of vertices and a set E of edges, each of which is an unordered pair $\{v, w\}$ of distinct vertices. A graph is *k-connected* if it has more than k vertices and remains connected whenever fewer than k vertices are removed. A graph class $\mathcal{G}$ is k-connected if every graph $G \in \mathcal{G}$ is k-connected. The *(open) neighborhood* 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 neighborhood* of v in G is in turn defined by $\mathcal{N}_G[v] \coloneqq \mathcal{N}_G(v) \cup \{v\}$. In addition, we denote $\mathcal{V}(G)$, the vertices set of graph G.

Isomorphically closed graph class. Following [BBC+20], in this work we will consider *isomorphically closed* graph classes.

Definition 2.1 (isomorphically closed)**.** *Let $n \in \mathbb{N}$ and let $\mathcal{G}$ be a graph class comprised of graphs with at most n vertices and labeled by a subset of $\{1, \ldots, n\}$. The graph class $\mathcal{G}$ is* isomorphically closed *if for every graph $G = (V, E) \in \mathcal{G}$ and for every bijection $\tau : V \to \{1, \ldots, n\}$, the graph $H = (\tau(V), E_H)$ is also in $\mathcal{G}$, where E_H is defined by $(u, v) \in E \Leftrightarrow (\tau(u), \tau(v)) \in E_H$.*

In other words, if a graph is in an isomorphically closed class, then so are all the graphs obtained by relabeling its vertices using a subset of $\{1, \ldots, n\}$. Furthermore, given an unlabeled graph G with n nodes, define *the graph class associated with* G as the graph class containing the labeled graphs $H = (\tau(V), E_H)$ where $\tau : V \to \{1, \ldots, n\}$ is a bijection and E_H is defined as above. By definition, the graph class associated with G is isomorphically closed.

2.1 Topology-Hiding Broadcast (THB)

Following [MOR15] we consider two definitions of topology-hiding broadcast. Our positive results (protocol constructions) are defined with respect to the stronger simulation-based definition while our lower bounds are given with respect to the weaker indistinguishability-based definition. Some of the content in this section is taking verbatim from [BBC+20].

UC framework. The simulation-based definition is defined in the UC framework of [Can01]; we present an informal overview of the model in Appendix A. Unless stated otherwise, we will consider computationally unbounded, static, and semi-honest adversaries and environments.

2.1.1 Simulation-Based THC

We recall the definition of simulation-based topology-hiding computation from [MOR15, BBC+19]. The real-world protocol is defined in a model where all communication is transmitted via the functionality $\mathcal{F}_{\mathsf{graph}}^{\mathcal{G}}$ (described in Figure 11). The functionality is parametrized by a family of graphs $\mathcal{G}$, representing all possible network topologies (aka communication graphs) that the protocol supports. We implicitly assume that every node in a graph is associated with a specific *party identifier*, pid. To simplify the notation, we will consider that P_v in the protocol is associated with node v in the graph.

Initially, before the protocol begins, $\mathcal{F}_{\mathsf{graph}}^{\mathcal{G}}$ receives the $\mathsf{network\ communication\ graph}$ G from a special graph party $\mathsf{P}_{\mathsf{graph}}$, makes sure that $G \in \mathcal{G}$, and provides to each party P_v with $v \in V$ its local neighbor-set. Next, during the protocol's execution, whenever party P_v wishes to send a message m to party P_w, it sends (v, w, m) to the functionality; the functionality verifies that the edge (v, w) is indeed in the graph, and if so delivers (v, w, m) to P_w.

Note that if all the graphs in $\mathcal{G}$ have exactly n nodes, then the exact number of participants is known to all and need not be kept hidden. In this case, defining the ideal functionality and constructing protocols becomes a simpler task. However, if there exist graphs in $\mathcal{G}$ that contain a *different* number of nodes, then the model must support functionalities and protocols that only know an *upper bound* on the number of participants. In the latter case, the actual number of participating parties must be kept hidden.

Given a class of graphs $\mathcal{G}$ with an upper bound n on the number of parties, we define a protocol π with respect to $\mathcal{G}$ as a set of n PPT interactive Turing machines (ITMs) $(\mathsf{P}_1, \ldots, \mathsf{P}_n)$ (the parties), where any subset of them may be activated with (potentially empty) inputs. Only the parties that have been activated participate in the protocol, send messages to one another (via $\mathcal{F}_{\mathsf{graph}}^{\mathcal{G}}$), and produce output.

An ideal-model computation of a functionality $\mathcal{F}$ is augmented to provide the corrupted parties with the information that is leaked about the graph; namely, every corrupted (dummy) party should learn its neighbor-set. Note that the functionality $\mathcal{F}$ can be completely agnostic about the actual graph that is used, and even about the family $\mathcal{G}$. To augment $\mathcal{F}$ in a generic way, we define the wrapper-functionality $\mathcal{W}_{\mathsf{graph\text{-}info}}^{\mathcal{G}}(\mathcal{F})$, that runs internally a copy of the functionality $\mathcal{F}$. The wrapper $\mathcal{W}_{\mathsf{graph\text{-}info}}^{\mathcal{G}}(\cdot)$ acts as a shell that is responsible to provide the relevant leakage to the corrupted parties; the original functionality $\mathcal{F}$ is the core that is responsible for the actual ideal computation.

More specifically, the wrapper receives the graph $G = (V, E)$ from the graph party $\mathsf{P}_{\mathsf{graph}}$, makes sure that $G \in \mathcal{G}$, and sends a special initialization message containing G to $\mathcal{F}$. (If the functionality $\mathcal{F}$ does not depend on the communication graph, it can ignore this message.) The wrapper then proceeds to process messages as follows: Upon receiving an initialization message from a party P_v responds with its neighbor set $\mathcal{N}_G(v)$ (just like $\mathcal{F}_{\mathsf{graph}}^{\mathcal{G}}$). All other input messages from a party P_v are forwarded to $\mathcal{F}$ and every message from $\mathcal{F}$ to a party P_v is delivered to its recipient.

Note that formally, the set of all possible parties V^* is fixed in advance. To represent a graph $G' = (V', E')$ where $V' \subseteq V^*$ is a subset of the parties, we use the graph $G = (V^*, E')$, where all vertices $v \in V^* \setminus V'$ have degree 0.

The functionality $\mathcal{F}^{\mathcal{G}}_{\mathsf{graph}}$

The functionality $\mathcal{F}^{\mathcal{G}}_{\mathsf{graph}}$ is parametrized by a family of graphs $\mathcal{G}$; let n denote the maximal number of nodes in $G \in \mathcal{G}$. The functionality proceeds with a special graph party $\mathsf{P}_{\mathsf{graph}}$ and with a subset of the parties $\mathsf{P}_1, \ldots, \mathsf{P}_n$ (to be defined by the graph received from $\mathsf{P}_{\mathsf{graph}}$) as follows.

Initialization Phase:

Input: $\mathcal{F}^{\mathcal{G}}_{\mathsf{graph}}$ waits to receive the graph $G = (V, E)$ from $\mathsf{P}_{\mathsf{graph}}$. If $G \notin \mathcal{G}$, abort.

Output: Upon receiving an initialization message from P_v, verify that $v \in V$, and if so send $\mathcal{N}_G(v)$ to P_v.

Communication Phase:

Input: $\mathcal{F}^{\mathcal{G}}_{\mathsf{graph}}$ receives from a party P_v a destination/data pair (w, m) where $w \in \mathcal{N}_G(v)$ and m is the message P_v wants to send to P_w. (If $v, w \notin V$, or if w is not a neighbor of v, $\mathcal{F}^{\mathcal{G}}_{\mathsf{graph}}$ ignores this input.)

Output: $\mathcal{F}^{\mathcal{G}}_{\mathsf{graph}}$ gives output (v, m) to P_w indicating that P_v sent the message m to P_w.

Figure 11: The communication graph functionality

Definition 2.2 (Topology-hiding computation)**.** *We say that a protocol π securely realizes a functionality $\mathcal{F}$ in a* topology-hiding manner *with respect to $\mathcal{G}$ tolerating a semi-honest adversary corrupting t parties if π securely realizes $\mathcal{W}^{\mathcal{G}}_{\mathsf{graph\text{-}info}}(\mathcal{F})$ in the $\mathcal{F}^{\mathcal{G}}_{\mathsf{graph}}$-hybrid model tolerating a semi-honest adversary corrupting t parties.*

Broadcast. In this work we will focus on topology-hiding computation of the *broadcast* functionality (see Figure 12), where a designated and publicly known party, named *the broadcaster*, starts with an input value m. Our broadcast functionality guarantees that every party that is connected to the broadcaster in the communication graph receives the message m as output, whereas parties in other connected components output adversarially chosen values (in particular, there is no "global agreement" requirement, but only in the connected component of the broadcaster). In this paper, we assume the communication graphs may have multiple connected components; however, there is only one "main" connected component and all other are singletons (having degree-0). The broadcaster may or may not be in the main connected component. Parties that are *not* connected to the broadcaster receive a message that is supplied by the adversary (we can consider stronger versions of broadcast, but this simplifies the proofs).

We denote the broadcast functionality where the broadcaster is P_v by $\mathcal{F}_{\mathsf{bc}}(\mathsf{P}_v)$.

Definition 2.3 (t-THB)**.** *Let $\mathcal{G}$ be a family of graphs and let t be an integer. A protocol π is a* t-THB protocol with respect to $\mathcal{G}$ *if $\pi(\mathsf{P}_v)$ securely realizes $\mathcal{F}_{\mathsf{bc}}(\mathsf{P}_v)$ in a topology-hiding manner with respect to $\mathcal{G}$, for every P_v, tolerating a semi-honest adversary corrupting t parties.*

The functionality $\mathcal{F}_{\mathsf{bc}}(\mathsf{P}_v)$

The broadcast functionality $\mathcal{F}_{\mathsf{bc}}(\mathsf{P}_v)$ is parametrized by the broadcaster P_v and proceeds as follows.

Initialization: The functionality receives the communication graph G from the wrapper $\mathcal{W}_{\mathsf{graph\text{-}info}}$.

Input: Record the input message $m \in \{0, 1\}$ sent by the broadcaster P_v.

Output: Send the output m to every party that is in the same connected component as P_v in G. For every other party in G, the output delivered to that party is supplied by the adversary.

Figure 12: The broadcast functionality

2.1.2 Indistinguishability-Based THC

Moran et al. [MOR15] gave a weaker definition of topology-hiding computation: IND-CTA security (indistinguishability under Chosen Topology Attack). We will next provide the explicit definitions for THB.

Definition 2.4 (1-IND-CTA THB)**.** *A broadcast protocol π is* indistinguishable under chosen topology attack against one semi-honest corruption (1-IND-CTA secure) *with respect to a graph class $\mathcal{G}$, if for any* PPT *adversary Adv there exists a negligible function* negl*, such that for every $\kappa \in \mathbb{N}$ it holds that*

$$\Pr\left[\mathsf{ExpTHB}^{\texttt{1-ind-cta}}_{\pi,\mathcal{G},\mathsf{Adv}}(\kappa) = 1\right] \leq 1/2 + \mathsf{negl}(\kappa),$$

where $\mathsf{ExpTHB}^{\texttt{1-ind-cta}}_{\pi,\mathcal{G},\mathsf{Adv}}(\kappa)$ is as defined in Figure 13 and the probability is taken over the random coins of the experiment and of the adversary.

The experiment $\mathsf{ExpTHB}^{\texttt{1-ind-cta}}_{\pi,\mathcal{G},\mathsf{Adv}}(\kappa)$

Choice phase. The challenger invokes Adv on input $(\mathcal{G}, 1^\kappa)$. Adv chooses two graphs $G_0, G_1 \in \mathcal{G}$, a broadcaster node $u \in V(G_0) \cap V(G_1)$, a message $m \in \{0, 1\}$, and a corrupted node $v \in V(G_0) \cap V(G_1)$ with $\mathcal{N}_{G_0}(v) = \mathcal{N}_{G_1}(v)$. Next, the adversary returns (G_0, G_1, u, m, v). If Adv's output is not of the required form, the experiment aborts.

Challenge phase. The challenger flips a random bit $b \leftarrow \{0, 1\}$ and runs $\pi(1^\kappa)$ with node u broadcasting message m over graph G_b, where the adversary controls node v.

Output phase. At the conclusion of the execution of π, Adv outputs b'. If $b = b'$, the experiment outputs 1; otherwise 0.

Figure 13: The 1-IND-CTA broadcast experiment

Definition 2.4 can be extended to support t corruptions, denoted t-IND-CTA broadcast, by having the adversary choose a set $I \subseteq [n]$ of size t satisfying $I \subseteq V(G_0) \cap V(G_1)$ in $\mathsf{ExpTHB}^{\texttt{1-ind-cta}}$, instead of choosing a single node v.

As shown in [MOR15], the indistinguishability-based definition is in fact implied by its simulation-based counterpart.

Proposition 2.5. *If π is 1-THB with respect to $\mathcal{G}$, then π is a 1-IND-CTA secure broadcast protocol with respect to $\mathcal{G}$.*

3 1-IT-THB for Friendship Graphs

In this section, we present the protocol for friendship graphs. We begin by defining the friendship graph class. Next, in Section 3.1, we present 1-THB for a fixed-size class. In Section 3.2, we extend the protocol to a variable-size class. Finally, in Section 3.3, we show that the protocol is in fact secure for any number of corruptions.

Definition 3.1 (friendship graph [ERS66])**.** *Let $n \in \mathbb{N}$. The* friendship graph F_n *is a planar, undirected graph with $2n+1$ nodes and $3n$ edges. Equivalently, F_n is the graph obtained by joining n triangles with a common center node.*

Definition 3.2 (friendship graph-class)**.** *Let $n \in \mathbb{N}$. The* friendship graph-class $\mathcal{G}_{\mathsf{friendship}}(n)$ *is the isomorphically closed graph class associated with F_n.*

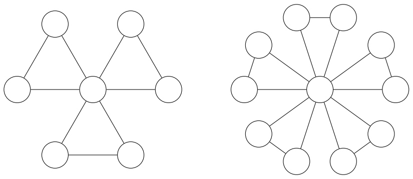


Figure 14: The friendship graphs F_3 on the left, and F_5 on the right.

We emphasize that in this section n stands for the number of triangles in the friendship graph F_n. The number of nodes (i.e., the number of participating parties) is $2n+1$.

3.1 1-IT-THB for Friendship Graphs of Fixed Size

We begin by presenting a protocol for $\mathcal{G}_{\mathsf{friendship}}(n)$ in which all graphs are of the same size. The protocol extends to support variable-size graphs with small adjustments; we mark down the additional instructions in blue.

The protocol $\pi_{\mathsf{friendship}}$, formally defined in Figure 15, operates in the $\mathcal{F}_{\mathsf{graph}}^{\mathcal{G}_{\mathsf{friendship}}(n)}$-hybrid model (see Section 2.1). Recall that this means that the communication in the protocol is carried out by an external trusted party (denoted $\mathcal{F}_{\mathsf{graph}}^{\mathcal{G}_{\mathsf{friendship}}(n)}$) that initially receives a graph G in the class $\mathcal{G}_{\mathsf{friendship}}(n)$. At the beginning of the protocol the trusted party hands each party its local neighbor set, and once a party v wants to send a message m to party u, the trusted party will do so if and only if the edge (u, v) is in G.

Notations used in the protocol. We refer the reader to Section 1.2.1 for an overview of the protocol. Below, we describe the notation used in the construction.

- The *blinding terms* are represented by the vectors $\boldsymbol{b}_u^{\mathsf{mul}}$ and $\boldsymbol{b}_u^{\mathsf{add}}$. These vectors consist of uniformly random values from $\mathbb{F}_q$, which are secret-shared between each party and its neighbors to hide the message share that is sent in the last step of the protocol from the neighbors of the receiver to the receiver.
- In case that the receiver is the center node, the *suitable offset* values $\beta_u^{\text{2-nbr}}$ are used to hide the structure of the graph. In this case, the center sends each of its neighbors a different

$\beta_u^{\text{2-nbr}}$, to ensure that within the message vector of the last step of the protocol, the values are either sampled uniformly at random or distributed as a uniformly sampled value.

- In the last step of the protocol, every neighbor u of the receiver sends to the receiver a vector $\boldsymbol{s}_u$. These vectors contains mainly uniformly sampled values, but also carefully planted shares of the message, to enable the receiver to reconstruct the message.

As mentioned above, the protocol $\pi_{\mathsf{friendship}}$ (see Figure 15) is suitable for a family of friendship graphs of variable size (as defined in Section 3.2) and, further, is secure against $t < n$ corruptions. We will prove this in a gradual manner, starting in Lemma 3.3 with a proof for a friendship graph with fixed-size and for $t = 1$. In this case, the blue instruction lines in Figure 15 are redundant and can be ignored.

Lemma 3.3 (IT-THB for friendship graph-class)**.** *Let $n \in \mathbb{N}$ with $n \geq 2$, and let $\mathcal{G}_{\mathsf{friendship}}(n)$ be the friendship graph-class with $2n+1$ nodes. The protocol $\pi_{\mathsf{friendship}}$ (defined in Figure 15) is a perfectly secure IT-THB protocol against a single semi-honest corruption with respect to $\mathcal{G}_{\mathsf{friendship}}(n)$.*

Proof. We will prove *correctness* and *security* separately.

Correctness. In case P_R is a neighbor of P_S, then in the first round P_R receives m and sets m as its output. In case P_R is *not* a neighbor of P_S, then P_R has two neighbors; denote $\mathcal{N}(R) = \{v, u\}$. By the structure of friendship graphs, P_R cannot be the center in this case, and one of its neighbors, say P_u, must be the center. The output in this case is defined to be

$$\begin{aligned}
&\boldsymbol{s}_v[u] \oplus \boldsymbol{s}_u[v] = \\
&= ((\beta_v^{\text{2-nbr}} \cdot (\boldsymbol{b}_v^{\mathsf{mul}}[u] \oplus \boldsymbol{b}_u^{\mathsf{mul}}[v])) \oplus (\boldsymbol{b}_u^{\mathsf{add}}[v] \oplus \boldsymbol{b}_v^{\mathsf{add}}[u]) \oplus m_v) \\
&\quad \oplus ((\beta_u^{\text{2-nbr}} \cdot (\boldsymbol{b}_u^{\mathsf{mul}}[v] \oplus \boldsymbol{b}_v^{\mathsf{mul}}[u])) \oplus (\boldsymbol{b}_v^{\mathsf{add}}[u] \oplus \boldsymbol{b}_u^{\mathsf{add}}[v]) \oplus m_u) \\
&= (\beta^{\text{2-nbr}} \cdot (\boldsymbol{b}_v^{\mathsf{mul}}[u] \oplus \boldsymbol{b}_u^{\mathsf{mul}}[v] \oplus \boldsymbol{b}_u^{\mathsf{mul}}[v] \oplus \boldsymbol{b}_v^{\mathsf{mul}}[u])) \\
&\quad \oplus (\boldsymbol{b}_u^{\mathsf{add}}[v] \oplus \boldsymbol{b}_v^{\mathsf{add}}[u] \oplus \boldsymbol{b}_v^{\mathsf{add}}[u] \oplus \boldsymbol{b}_u^{\mathsf{add}}[v]) \oplus (m_v \oplus m_u) \\
&= (\beta^{\text{2-nbr}} \cdot 0) \oplus (0 \oplus 0) \oplus (0 \oplus m) = m,
\end{aligned}$$

where the first equality holds by the way $\boldsymbol{s}_v[u]$ and $\boldsymbol{s}_u[v]$ are defined in Step 4; the second equality holds by Step 3 and because the receiver P_R, which is not the center, sets $\beta_v^{\text{2-nbr}} = \beta_u^{\text{2-nbr}} = \beta^{\text{2-nbr}}$; and the last equality holds because the non-center neighbor P_v sets $m_v = 0$ in Step 4.

Security. We proceed to prove security. Let P_{v^*} with $v^* \in [2n+1]$ denote the corrupted party. We will construct a simulator Sim that given the neighbor-set of P_{v^*} generates a simulated view for P_{v^*} that is identically distributed as its view in a real execution of the protocol. As we consider semi-honest security, and as broadcast is a deterministic functionality, this implies that the environment's output is identically distributed in the real and ideal computations.

The simulator Sim runs P_{v^*} in its head. Initially, P_{v^*} sends an initialization message to $\mathcal{F}_{\mathsf{graph}}^{\mathcal{G}}$; the simulator forwards this message to $\mathcal{W}_{\mathsf{graph\text{-}info}}^{\mathcal{G}}(\mathcal{F}_{\mathsf{bc}}(\mathsf{P}_S))$ and sends the response $\mathcal{N}(v^*)$ to P_{v^*}. Next, in case the sender is corrupted, i.e., $\mathsf{P}_{v^*} = \mathsf{P}_S$, the simulator sends its input m to $\mathcal{W}_{\mathsf{graph\text{-}info}}^{\mathcal{G}}(\mathcal{F}_{\mathsf{bc}}(\mathsf{P}_S))$; regardless, Sim receives the message m as the output from $\mathcal{W}_{\mathsf{graph\text{-}info}}^{\mathcal{G}}(\mathcal{F}_{\mathsf{bc}}(\mathsf{P}_S))$ (formally, Sim sends the "empty input" for a non-sender corrupted party).

Protocol $\pi_{\mathsf{friendship}}(n, \mathsf{P}_S)$

Auxiliary input: A binary field $\mathbb{F}_q$ such that $q > 2n$.

Input: The sender P_S, with $S \in [2n+1]$, holds an input $m \in \{0,1\}$.

Hybrid model: The protocol is defined in the $\mathcal{F}_{\mathsf{graph}}^{\mathcal{G}_{\mathsf{friendship}}(n)}$-hybrid model.

The protocol:

- Each party P_v sends an initialization message to $\mathcal{F}_{\mathsf{graph}}^{\mathcal{G}_{\mathsf{friendship}}(n)}$ and receives its neighbor-set $\mathcal{N}(v)$.
- Repeat for each potential receiver P_R with $R \in [2n+1]$:

1. *Forwarding the message to the center node.* The sender P_S sends m to its neighbors.
   If P_S is a isolated, the center acts as if received $m = 0$ from P_S.
2. *Generating blinding terms.* Every non-receiver $\mathsf{P}_u \neq \mathsf{P}_R$ uniformly samples from $\mathbb{F}_q^{2n+1}$ two blinding vectors $\boldsymbol{b}_u^{\mathsf{mul}}$ and $\boldsymbol{b}_u^{\mathsf{add}}$ of size $2n+1$, and sends $(\boldsymbol{b}_u^{\mathsf{mul}}[v], \boldsymbol{b}_u^{\mathsf{add}}[v])$ to every P_v with $v \in \mathcal{N}(u)$.
   In case P_u is the center, it plays in its head every isolated P_v; that is, P_u samples on behalf of P_v two blinding vectors $\boldsymbol{b}_v^{\mathsf{mul}}$ and $\boldsymbol{b}_v^{\mathsf{add}}$ and sends $(\boldsymbol{b}_v^{\mathsf{mul}}[u], \boldsymbol{b}_v^{\mathsf{add}}[u])$ to itself.
3. *Generating suitable offsets.* Party P_R generates values $\beta_u^{\text{2-nbr}}$ for each of its neighbors $u \in \mathcal{N}(R)$, and sends $\beta_u^{\text{2-nbr}}$ to P_u, as follows:
   - If P_R is the center, sample independently at random $\beta_u^{\text{2-nbr}} \leftarrow \mathbb{F}_q$, for each $u \in \mathcal{N}(R)$, conditioned on all values being distinct.
   - Otherwise, sample $\beta^{\text{2-nbr}} \leftarrow \mathbb{F}_q$ and set $\beta_u^{\text{2-nbr}} := \beta^{\text{2-nbr}}$ for every $u \in \mathcal{N}(R)$.
4. *Forwarding m to P_R.* In this step, only parties P_u with $u \in \mathcal{N}(R)$ send messages, and only P_R receives messages. P_u initializes a vector of zeros $\boldsymbol{s}_u$ of size $2n+1$. For every $v \in [2n+1] \setminus \{R, u\}$, if P_u is not the center and $v \notin \mathcal{N}(u)$, set $\boldsymbol{s}_u[v] \leftarrow \mathbb{F}_q$ to be a random value; otherwise, set

   $$\boldsymbol{s}_u[v] := (\beta_u^{\text{2-nbr}} \cdot (\boldsymbol{b}_u^{\mathsf{mul}}[v] \oplus \boldsymbol{b}_v^{\mathsf{mul}}[u])) \oplus (\boldsymbol{b}_u^{\mathsf{add}}[v] \oplus \boldsymbol{b}_v^{\mathsf{add}}[u]) \oplus m_u,$$

   where $m_u := m$ if P_u is the center party, and $m_u := 0$ otherwise.
   Finally, P_u sends to P_R the vector $\boldsymbol{s}_u$.
5. P_R *output.*
   - If P_R is P_S then output its input m.
   - If $R \in \mathcal{N}(S)$, output m as received on Step 1.
   - If $|\mathcal{N}(R)| = 2$, denote $\mathcal{N}(R) = \{v_1, v_2\}$ and output $\boldsymbol{s}_{v_1}[v_2] \oplus \boldsymbol{s}_{v_2}[v_1]$.
   - If $|\mathcal{N}(R)| = 0$, output 0.

Figure 15: Information-theoretic THB over $\mathcal{G}_{\mathsf{friendship}}(n)$. The instruction lines in blue are only needed when considering friendship graphs with variable size (i.e., when some nodes are isolated).

Next, Sim simulates each iteration of the protocol towards the corrupted party, for every potential receiver P_R with $R \in [2n+1] \setminus \{S\}$.

- To simulate Step 1, if the sender is corrupted, Sim receives m on behalf of every honest P_v for which $v \in \mathcal{N}(S)$. If the sender is honest and $S \in \mathcal{N}(v^*)$ then Sim sends m to P_{v^*}.
- To simulate Step 2, if P_{v^*} is not the receiver, Sim samples for every honest neighbor P_u with $u \in \mathcal{N}(v^*)$ random blinding terms $\boldsymbol{b}_u^{\mathsf{mul}}[v^*], \boldsymbol{b}_u^{\mathsf{add}}[v^*] \leftarrow \mathbb{F}_q$ and sends $(\boldsymbol{b}_u^{\mathsf{mul}}[v^*], \boldsymbol{b}_u^{\mathsf{add}}[v^*])$ to P_{v^*} on behalf of P_u. In addition, Sim receives $(\boldsymbol{b}_{v^*}^{\mathsf{mul}}[u], \boldsymbol{b}_{v^*}^{\mathsf{add}}[u])$ from P_{v^*} on behalf of P_u.

- To simulate Step 3, if P_{v^*} is a neighbor of P_R, i.e., $R \in \mathcal{N}(v^*)$, the simulator samples at random $\beta^{\text{2-nbr}}_{v^*} \leftarrow \mathbb{F}_q$ and sends $\beta^{\text{2-nbr}}_{v^*}$ to P_{v^*}. Else, If $v^* = R$, the simulator receives $\beta^{\text{2-nbr}}_{u}$ from P_{v^*} on behalf of P_u for every $u \in \mathcal{N}(v^*)$.
- To simulate Step 4, if P_{v^*} is a neighbor of the receiver, i.e., $v^* \in \mathcal{N}(R)$, then receive from P_{v^*} the vector $\boldsymbol{s}_{v^*}$ on behalf of P_R.

  If P_{v^*} is the receiver, we consider two cases:

  1. If P_{v^*} is the center node, then for each $u \neq v^*$ and for every $v \notin \{u, v^*\}$, the simulator samples a vector $\boldsymbol{s}_u$ from $\mathbb{F}_q^{2n+1}$ at random and sends it to P_{v^*} on behalf of P_u.
  2. Otherwise, denote $\mathcal{N}(v^*) = \{w_1, w_2\}$. The simulator initializes two vectors $\boldsymbol{s}_{w_1}$ and $\boldsymbol{s}_{w_2}$, each of size $2n + 1$, with zeros. The values $\boldsymbol{s}_{w_1}[w_2]$ and $\boldsymbol{s}_{w_2}[w_1]$ are sampled from $\mathbb{F}_q$ conditioned on $\boldsymbol{s}_{w_1}[w_2] \oplus \boldsymbol{s}_{w_2}[w_1] = m$. Sim samples $\boldsymbol{s}_{w_1}[v], \boldsymbol{s}_{w_2}[v] \leftarrow \mathbb{F}_q$ at random for every $v \in [2n + 1] \setminus \{v^*, w_1, w_2\}$. Finally, Sim sends to the adversary the vectors $\boldsymbol{s}_{w_1}$ and $\boldsymbol{s}_{w_2}$.

We proceed to show that the view of P_{v^*} in the simulated protocol is identically distributed as its view in a real execution of the protocol. Note that the simulation mirrors the protocol behavior except for the last step. Therefore, we need to analyze only the last step, where v^* is the receiver.

Recall that P_{v^*} receives in Step 4 from each of its neighbors P_w, with $w \in \mathcal{N}(v^*)$, the vector $\boldsymbol{s}_w$ that is computed as follows: if $v \in \mathcal{N}(w)$ the value $\boldsymbol{s}_w[v]$ is set to be

$$\boldsymbol{s}_w[v] := (\beta^{\text{2-nbr}}_w \cdot (\boldsymbol{b}^{\mathsf{mul}}_w[v] \oplus \boldsymbol{b}^{\mathsf{mul}}_v[w])) \oplus (\boldsymbol{b}^{\mathsf{add}}_w[v] \oplus \boldsymbol{b}^{\mathsf{add}}_v[w]) \oplus m_w,$$

else $\boldsymbol{s}_w[v]$ is sampled uniformly at random.

We will prove separately the cases where the corrupted party is the center and where it is not.

Case 1: P_{v^*} is not the center, i.e., has 2 neighbors. Denote $\mathcal{N}(v^*) = \{w_1, w_2\}$, since one of the neighbors is the center, in the real execution it holds that $\boldsymbol{s}_{w_1}[w_2] \oplus \boldsymbol{s}_{w_2}[w_1] = m$. Indeed, the simulation guarantees all values are sampled conditioned on $\boldsymbol{s}_{w_1}[w_2] \oplus \boldsymbol{s}_{w_2}[w_1] = m$. The joint view is identically distributed because for every $v \notin \mathcal{N}[v^*]$ and every $i \in \{1, 2\}$ the values $\boldsymbol{s}_{w_i}[v]$ are formed as follows: if $v \notin \mathcal{N}(w_i)$ then $\boldsymbol{s}_{w_i}[v] \leftarrow \mathbb{F}_q$; otherwise, for $v \in \mathcal{N}(w_i)$,

$$\boldsymbol{s}_{w_i}[v] = (\beta^{\text{2-nbr}}_{w_i} \cdot (\boldsymbol{b}^{\mathsf{mul}}_{w_i}[v] \oplus \boldsymbol{b}^{\mathsf{mul}}_v[w_i])) \oplus (\boldsymbol{b}^{\mathsf{add}}_{w_i}[v] \oplus \boldsymbol{b}^{\mathsf{add}}_v[w_i]) \oplus m_{w_i},$$

where $\boldsymbol{b}^{\mathsf{add}}_{w_i}[v]$ and $\boldsymbol{b}^{\mathsf{add}}_v[w_i]$ are sampled uniformly at random, and are only known to P_{w_i} and P_v. Note that $v \notin \mathcal{N}[v^*]$ therefore $v \notin \{w_1, w_2\}$. Particularly, P_{v^*} is not aware of those blinding terms, and each term is used only once.

Case 2: P_{v^*} is the center. Denote $\{w_1, w_2\} \subseteq \mathcal{N}(v^*)$. In both the real and the simulated executions, for $i \in \{1, 2\}$, P_{v^*} receives from w_i a vector $\boldsymbol{s}_{w_i}$. In the simulation, each $\boldsymbol{s}_{w_i}[v]$ is sampled uniformly at random for every $v \notin \{R, w_i\}$ and set to zero for $v \in \{R, w_i\}$. In the real execution if $v \notin \mathcal{N}(w_i)$ then $\boldsymbol{s}_{w_i}[v]$ is indeed sampled uniformly at random, but otherwise, it is formed by

$$\begin{aligned}\boldsymbol{s}_{w_i}[v] &= (\beta^{\text{2-nbr}}_{w_i} \cdot (\boldsymbol{b}^{\mathsf{mul}}_{w_i}[v] \oplus \boldsymbol{b}^{\mathsf{mul}}_v[w_i])) \oplus (\boldsymbol{b}^{\mathsf{add}}_{w_i}[v] \oplus \boldsymbol{b}^{\mathsf{add}}_v[w_i]) \oplus m_{w_i} \\ &= (\beta^{\text{2-nbr}}_{w_i} \cdot (\boldsymbol{b}^{\mathsf{mul}}_{w_i}[v] \oplus \boldsymbol{b}^{\mathsf{mul}}_v[w_i])) \oplus (\boldsymbol{b}^{\mathsf{add}}_{w_i}[v] \oplus \boldsymbol{b}^{\mathsf{add}}_v[w_i]).\end{aligned}$$

The equality holds since P_{v^*} is the center, so each of its neighbors sets $m_{w_i} = 0$. Thus, it is required to show that if v is neighbor of w_i the value $\boldsymbol{s}_{w_i}[v]$ is distributed uniformly at random from the view of P_{v^*} (as done in the simulation). Differently from Case 1, we note that when w_1 and w_2 are neighbors, they use the same values to form $\boldsymbol{s}_{w_1}[w_2]$ and $\boldsymbol{s}_{w_2}[w_1]$

$$\boldsymbol{s}_{w_1}[w_2] = (\beta^{\text{2-nbr}}_{w_1} \cdot (\boldsymbol{b}^{\mathsf{mul}}_{w_1}[w_2] \oplus \boldsymbol{b}^{\mathsf{mul}}_{w_2}[w_1])) \oplus (\boldsymbol{b}^{\mathsf{add}}_{w_1}[w_2] \oplus \boldsymbol{b}^{\mathsf{add}}_{w_2}[w_1])$$
$$\boldsymbol{s}_{w_2}[w_1] = (\beta^{\text{2-nbr}}_{w_2} \cdot (\boldsymbol{b}^{\mathsf{mul}}_{w_2}[w_1] \oplus \boldsymbol{b}^{\mathsf{mul}}_{w_1}[w_2])) \oplus (\boldsymbol{b}^{\mathsf{add}}_{w_2}[w_1] \oplus \boldsymbol{b}^{\mathsf{add}}_{w_1}[w_2]).$$

In the view of P_{v^*}, the values $\boldsymbol{s}_{w_1}[w_2]$ and $\boldsymbol{s}_{w_2}[w_1]$ are the only values that are based on $\boldsymbol{b}^{\mathsf{mul}}_{w_1}[w_2]$, $\boldsymbol{b}^{\mathsf{mul}}_{w_2}[w_1]$, $\boldsymbol{b}^{\mathsf{add}}_{w_1}[w_2]$, and $\boldsymbol{b}^{\mathsf{add}}_{w_2}[w_1]$, which in turn are each sampled independently and uniformly at random, and remain unknown to P_{v^*}. In addition, note that the values $\beta^{\text{2-nbr}}_{w_1}$ and $\beta^{\text{2-nbr}}_{w_2}$ are *known* to P_{v^*} (as P_{v^*} sampled and sent them), and by Step 3 we are guaranteed that $\beta^{\text{2-nbr}}_{w_1} \neq \beta^{\text{2-nbr}}_{w_2}$ because P_{v^*} is the center. Denote,

$$b^{\mathsf{mul}} = \boldsymbol{b}^{\mathsf{mul}}_{w_1}[w_2] \oplus \boldsymbol{b}^{\mathsf{mul}}_{w_2}[w_1] \quad \text{and} \quad b^{\mathsf{add}} = \boldsymbol{b}^{\mathsf{add}}_{w_2}[w_1] \oplus \boldsymbol{b}^{\mathsf{add}}_{w_1}[w_2].$$

Note that b^{mul} and b^{add} are uniformly distributed. Looking back at $\boldsymbol{s}_{w_1}[w_2]$ and $\boldsymbol{s}_{w_2}[w_1]$, it holds that

$$\boldsymbol{s}_{w_1}[w_2] = (\beta^{\text{2-nbr}}_{w_1} \cdot b^{\mathsf{mul}}) \oplus b^{\mathsf{add}} \quad \text{and} \quad \boldsymbol{s}_{w_2}[w_1] = (\beta^{\text{2-nbr}}_{w_2} \cdot b^{\mathsf{mul}}) \oplus b^{\mathsf{add}}.$$

From the point of view of P_{v^*}, these are two linear equations with two unknowns b^{mul} and b^{add}, which are uniformly distributed. These equations are solvable since $\beta^{\text{2-nbr}}_{w_1} \neq \beta^{\text{2-nbr}}_{w_2}$. Therefore, from the view of P_{v^*}, the values $\boldsymbol{s}_{w_1}[w_2]$ and $\boldsymbol{s}_{w_2}[w_1]$ are uniformly distributed.

This concludes the proof of Lemma 3.3. □

3.2 1-IT-THB for Friendship Graphs of Variable Size

In the previous section, we considered a friendship graph of a fixed and known size, and showed that the protocol $\pi_{\mathsf{friendship}}$ is secure against a single corruption. Note that in this protocol the only party whose actions depend on the number of nodes is the center (that already knows this information), whereas non-center nodes only need to know an upper bound on the size of the graph.

We proceed to use this intuition and show that the protocol $\pi_{\mathsf{friendship}}$ provides security when the graph class includes various friendship graphs of different sizes. Recall that hiding the number of nodes in the graph may raise new challenges, e.g., THB with sender-anonymity for the class of 2-paths and 3-paths was shown to imply infinitely often oblivious transfer [BBC+20].

In this setting, we consider the protocol $\pi_{\mathsf{friendship}}$ *including* the instructions in blue. Recall that the definition of THB (Definition 2.2) considers a known upper bound on the number of parties, but only guarantees agreement and validity for parties that are on the same connected component as the sender. When considering friendship graphs with $2k+1$ nodes, for $k < n$, we view nodes that are not on the same connected component of the sender as singletons (isolated parties), that always output 0 in the protocol. We begin by defining "augmented" friendship graphs that may include additional isolated nodes.

Definition 3.4 (augmented friendship graph)**.** *Let $k, n \in \mathbb{N}$ such that $k \leq n$. The* augmented friendship graph $F_{k,n}$ *consists of the friendship graph F_k together with $2n+1-(2k+1) = 2(n-k)$ isolated nodes.*

Definition 3.5 (augmented friendship graph-class)**.** *Let $k, n \in \mathbb{N}$ such that $k \leq n$. The* augmented friendship graph-class*, denoted by $\mathcal{G}_{\mathsf{friendship}}(k, n)$, is the isomorphically closed graph class associated with $F_{k,n}$.*

In our proof we will consider the union of augmented friendship graph-class $\bigcup_{k=2}^{n} \mathcal{G}_{\mathsf{friendship}}(k, n)$, such that the main connected component (of non-isolated nodes) is of variable size: namely, of possible size $2k + 1$ for any $k \leq n$.

Lemma 3.6 ($\mathsf{IT\text{-}THB}$ for friendship graph-class of variable size)**.** *Let $n \in \mathbb{N}$ with $n \geq 2$, and let $\mathcal{G} \subseteq \bigcup_{k=2}^{n} \mathcal{G}_{\mathsf{friendship}}(k, n)$. The protocol $\pi_{\mathsf{friendship}}$ (defined in Figure 15, including the blue instruction lines) is a perfectly secure $\mathsf{IT\text{-}THB}$ protocol against a single semi-honest corruption with respect to $\mathcal{G}$.*

Proof. The proof of correctness follows identically as the proof of Lemma 3.3, with the sole exception that every party outside of the connected component of the sender outputs 0. It is left to show that except for the center, no party identifies the specific graph that the protocol runs on.

Security. Let P_{v^*} with $v^* \in [2n + 1]$ denote the corrupted party. We will construct a simulator $\mathsf{Sim}_{\mathsf{var\text{-}size}}$ that given the neighbor-set of P_{v^*} generates a simulated view for P_{v^*} that is identically distributed as its view in a real execution of the protocol. As we consider semi-honest security, and as broadcast is a deterministic functionality, this implies that the environment's output is identically distributed in the real and ideal computations.

The simulator that will be constructed is very similar to the simulator in the proof of Lemma 3.3, with the three following differences. The first difference is that the simulator instructs $\mathcal{W}^{\mathcal{G}}_{\mathsf{graph\text{-}info}}(\mathcal{F}_{\mathsf{bc}}(\mathsf{P}_S))$ to set the output of every party that is not in the connected component of P_S to be 0. The second difference is when the sender is isolated, the simulator will emulate as if the sender has sent input 0. The third difference is when parties are isolated they will not receive or produce any messages. It is left to show that the view of P_{v^*} in the simulated protocol is identically distributed as its view in a real execution of the protocol. This is shown by considering the following cases:

Case 0: P_{v^*} is a singleton. In this case, no messages are received or sent; hence, the view is trivially identical.

Case 1: P_{v^*} has 2 neighbors. In this case, $\mathsf{Sim}_{\mathsf{var\text{-}size}}$ will run exactly like Sim on the fixed-sized friendship graph $\mathcal{G}_{\mathsf{friendship}}(n)$. Indeed, in the real execution the center "runs in its head" all isolated parties, which means that the view of P_{v^*} in the real execution is identically distributed as in the case of the fixed-size friendship graph F_n (where in case the sender is isolated, it is emulated in the fixed-sized friendship with input 0). The proof thus follows from Lemma 3.3.

Case 2: P_{v^*} is the center. We will show that the center's view is identically distributed as its view in the fixed-size case (the proof of Lemma 3.3) with respect to all the nodes that are connected to the center; we will then conclude that its view is identically distributed as in the real execution.

Consider the case where there are no isolated parties; in this case, the simulator $\mathsf{Sim}_{\mathsf{var\text{-}size}}$ run exactly like Sim on $\mathcal{G}_{\mathsf{friendship}}(n)$, and the proof follows from Lemma 3.3.

Otherwise, there are isolated parties; since P_{v^*} is the center the simulator can identify all isolated parties from the neighbor-set it receives from $\mathcal{W}^{\mathcal{G}}_{\mathsf{graph\text{-}info}}(\mathcal{F}_{\mathsf{bc}}(\mathsf{P}_S))$. The simulator runs Sim on $\mathcal{G}_{\mathsf{friendship}}(n)$, but emulates only the non-isolated parties. In the case the sender is isolated, the simulator $\mathsf{Sim}_{\mathsf{var\text{-}size}}$ runs exactly like Sim with input 0, and since the simulator acts the same as the real execution the view is identically distributed. For each $w \in \mathcal{N}(v^*)$ in both real and simulated execution, at the last step the center receives $\boldsymbol{s}_w$. In the simulated view the values in $\boldsymbol{s}_w$ are sampled uniformly at random. In the real execution, for values in $\boldsymbol{s}_w$ that correspond to non-neighbors of w, the value is sampled uniformly at random. For every $v \in \mathcal{N}(w)$ the value $\boldsymbol{s}_w[v]$ is defined as

$$\boldsymbol{s}_w[v] := (\beta_w^{\text{2-nbr}} \cdot (\boldsymbol{b}_w^{\mathsf{mul}}[v] \oplus \boldsymbol{b}_v^{\mathsf{mul}}[w])) \oplus (\boldsymbol{b}_w^{\mathsf{add}}[v] \oplus \boldsymbol{b}_v^{\mathsf{add}}[w]).$$

From Case 2 in the proof of Lemma 3.3 it follows that those values are uniformly distributed. Since the messages are uniformly distributed in both executions the view is identically distributed. □

3.3 Friendship Graphs: Beyond a Single Corruption

In this section, we present the first non-trivial feasibility of $\mathsf{IT\text{-}THB}$ secure against more than one corruption. In fact, we show that the protocol $\pi_{\mathsf{friendship}}$ is already secure against any number of corruptions. Stated differently, the joint view of any subset of nodes in the protocol does not reveal any new information about the connectivity of the remaining nodes, except from information that can be deduced from the local neighbor-sets of the colluding nodes.

Theorem 3.7. *Let $n \in \mathbb{N}$ with $n \geq 2$, let $t < 2n+1$, and consider the graph-class $\mathcal{G} \subseteq \bigcup_{k=2}^{n} \mathcal{G}_{\mathsf{friendship}}(k,n)$. The protocol $\pi_{\mathsf{friendship}}$ (defined in Figure 15, including the blue instructions) is a perfectly secure $\mathsf{IT\text{-}THB}$ protocol against t semi-honest corruptions with respect to $\mathcal{G}$.*

Proof. The proof of correctness follows identically as the proof of Lemma 3.6. Let $\mathcal{C} \subseteq [2n+1]$ denote the set of corrupted parties. We will construct a simulator Sim that given the neighbor-sets of nodes in $\mathcal{C}$ generates a simulated view for the corrupted parties that is identically distributed as their view in a real execution of the protocol. As we consider semi-honest security, and as broadcast is a deterministic functionality, this implies that the environment's output is identically distributed in the real and ideal computations.

The simulator Sim runs the corrupted parties in its head. Initially, every P_v with $v \in \mathcal{C}$ sends an initialization message to $\mathcal{F}_{\mathsf{graph}}^{\mathcal{G}}$; the simulator forwards this message to $\mathcal{W}_{\mathsf{graph\text{-}info}}^{\mathcal{G}}(\mathcal{F}_{\mathsf{bc}}(\mathsf{P}_S))$ and sends the response $\mathcal{N}(v)$ to P_v. Next, in case the sender is corrupted, i.e., $S \in \mathcal{C}$, the simulator sends its input m to $\mathcal{W}_{\mathsf{graph\text{-}info}}^{\mathcal{G}}(\mathcal{F}_{\mathsf{bc}}(\mathsf{P}_S))$; regardless, Sim receives the message m as the output from $\mathcal{W}_{\mathsf{graph\text{-}info}}^{\mathcal{G}}(\mathcal{F}_{\mathsf{bc}}(\mathsf{P}_S))$ (formally, Sim sends the "empty input" for every non-sender corrupted party). Further, the simulator instructs $\mathcal{W}_{\mathsf{graph\text{-}info}}^{\mathcal{G}}(\mathcal{F}_{\mathsf{bc}}(\mathsf{P}_S))$ to set the output of every party that is not in the connected component of P_S to be 0.

Next, Sim simulates an iteration of the protocol towards the corrupted parties, for every potential receiver P_R with $R \in [2n+1] \setminus \{S\}$.

- To simulate Step 1, if the sender is corrupted, Sim receives m on behalf of every honest P_v for which $v \in \mathcal{N}(S)$. If the sender is honest, Sim sends m to every corrupted P_{v^*} for which $S \in \mathcal{N}(v^*)$.

- To simulate Step 2, for every non-receiver corrupted party P_{v^*} with $v^* \in \mathcal{C} \setminus \{R\}$, the simulator samples for every honest neighbor P_u with $u \in \mathcal{N}(v^*) \setminus \mathcal{C}$ random blinding vectors $\boldsymbol{b}_u^{\mathsf{mul}}, \boldsymbol{b}_u^{\mathsf{add}}$ from

$\mathbb{F}_q^{2n+1}$ and sends the tuple $(\boldsymbol{b}_u^{\mathsf{mul}}[v^*], \boldsymbol{b}_u^{\mathsf{add}}[v^*])$ to P_{v^*} on behalf of P_u. In addition, Sim receives the tuple $(\boldsymbol{b}_{v^*}^{\mathsf{mul}}[u], \boldsymbol{b}_{v^*}^{\mathsf{add}}[u])$ from P_{v^*} on behalf of P_u.

- To simulate Step 3, if the receiver P_R is honest, let r denote the number of corrupted parties who *are* neighbors of P_R, i.e., $r = |\{v^* \mid v^* \in \mathcal{C} \wedge R \in \mathcal{N}(v^*)\}|$. Note that when $r = 2$ the simulation identifies whether the receiver is the center.

  If $r = 0$, then the receiver is not a neighbor of any corrupted party then no messages are received or sent in this step.

  If $r = 1$, then there exists a unique corrupted P_{v^*} who is a neighbor of P_R, i.e., $R \in \mathcal{N}(v^*)$, the simulator samples at random $\beta^{\text{2-nbr}} \leftarrow \mathbb{F}_q$ and sends $\beta^{\text{2-nbr}}$ to P_{v^*} on behalf of P_R.

  If $r = 2$ and the receiver is center, then for every corrupted party P_{v^*} who is a neighbor of P_R, i.e., $R \in \mathcal{N}(v^*)$, the simulator samples at random $\beta_{v^*}^{\text{2-nbr}} \leftarrow \mathbb{F}_q$ and sends $\beta_{v^*}^{\text{2-nbr}}$ to P_{v^*} on behalf of P_R.

  If $r = 2$ and the receiver is not the center, then for every corrupted party P_{v^*} who is a neighbor of P_R, i.e., $R \in \mathcal{N}(v^*)$, the simulator samples at random $\beta^{\text{2-nbr}} \leftarrow \mathbb{F}_q$ and sends $\beta^{\text{2-nbr}}$ to P_{v^*} on behalf of P_R.

  If the receiver P_R is corrupted, the simulator receives $\beta_u^{\text{2-nbr}}$ from P_R on behalf of P_u for every $u \in \mathcal{N}(R) \setminus \mathcal{C}$. Note that if the receiver is the center then all $\beta_u^{\text{2-nbr}}$ are different, else equals.

- To simulate Step 4, if the receiver P_R is honest, then for every corrupted party P_{v^*} who is a neighbor of P_R, i.e., with $R \in \mathcal{N}(v^*)$, the simulator receives on behalf of P_R from P_{v^*} the vector $\boldsymbol{s}_{v^*}$.

  If the receiver P_R is corrupted, we consider the following cases:

  If P_R is a singleton: then the simulator does not receive any messages on behalf of P_R.

  If P_R is not the center:

  1. If $|\mathcal{N}(R) \cap \mathcal{C}| = 0$, denote $\mathcal{N}(R) = \{w_1, w_2\}$. The simulator initializes two vectors $\boldsymbol{s}_{w_1}$ and $\boldsymbol{s}_{w_2}$ each of size $2n+1$ with zeros. It samples from $\mathbb{F}_q$ and sets the values $\boldsymbol{s}_{w_1}[w_2]$, $\boldsymbol{s}_{w_2}[w_1]$ conditioned on $\boldsymbol{s}_{w_1}[w_2] \oplus \boldsymbol{s}_{w_2}[w_1] = m$. For every $v \in [2n+1] \setminus \{v^*, w_1, w_2\}$ the simulator samples $\boldsymbol{s}_{w_1}[v], \boldsymbol{s}_{w_2}[v] \leftarrow \mathbb{F}_q$ at random and sends to the adversary the vectors $\boldsymbol{s}_{w_i}$ for $i \in \{1, 2\}$.
  2. If $|\mathcal{N}(R) \cap \mathcal{C}| = 1$ let $v^* \in \mathcal{N}(R) \cap \mathcal{C}$. Let $u \in \mathcal{N}(R) \cap \mathcal{N}(v)$. Party u is an honest neighbor of P_R. Note that P_R and P_{v^*} have a single common neighbor due to the friendship-graph definition. The simulator initializes the vector $\boldsymbol{s}_u$ to zeros, and for every $w \notin \{v^*, u, R\}$ sets $\boldsymbol{s}_u[w]$ to be a uniformly random value from $\mathbb{F}_q$. For $w = v^*$, it holds that
  $$\boldsymbol{s}_u[w] = \boldsymbol{s}_u[v^*] = (\beta_u \cdot (\boldsymbol{b}_u^{\mathsf{mul}}[v^*] \oplus \boldsymbol{b}_{v^*}^{\mathsf{mul}}[u])) \oplus (\boldsymbol{b}_u^{\mathsf{add}}[v^*] \oplus \boldsymbol{b}_{v^*}^{\mathsf{add}}[u]) \oplus m_u,$$
  where the values β_u, $\boldsymbol{b}_u^{\mathsf{mul}}[v^*]$, $\boldsymbol{b}_{v^*}^{\mathsf{mul}}[u]$, $\boldsymbol{b}_u^{\mathsf{add}}[v^*]$ and $\boldsymbol{b}_{v^*}^{\mathsf{add}}[u]$ are the sampled by the simulator in the previous steps, and $m_u = 0$ if P_{v^*} is the center, else $m_u = m$. Eventually, the simulator sends the vector $\boldsymbol{s}_u$ to P_R on behalf of P_u.
  3. If $|\mathcal{N}(R) \cap \mathcal{C}| = 2$, in this case no messages are sent to (and from) the simulator.

If P_R is the center:

1. If $|\mathcal{N}(R) \cap \mathcal{C}| = 0$. The simulator acts the same as in the proof Lemma 3.3. The simulator initializes a vector $\boldsymbol{s}_u$ of size $2n+1$ to zeros; next, for every $u \notin \mathcal{C}$ and every $v \notin \{u, R\}$, the simulator samples $\boldsymbol{s}_u[v] \leftarrow \mathbb{F}_q$ at random and sends $\boldsymbol{s}_u$ to P_R on behalf of P_u.
2. If $|\mathcal{N}(R) \cap \mathcal{C}| > 0$. Let $k = |\mathcal{N}(R) \cap \mathcal{C}|$ and denote $\mathcal{N}(R) \cap \mathcal{C} = \{v_1^*, v_2^*, \ldots, v_k^*\}$. For each $u \in \mathcal{N}(R) \setminus \mathcal{C}$, the simulator initializes a vector $\boldsymbol{s}_u$ of size $2n+1$ to zeros; note that if there exists $i \in [m]$ such that $u \in \mathcal{N}(v_i^*)$ then $\boldsymbol{s}_u[v_i^*]$ is formed as
$$\boldsymbol{s}_u[v_i^*] = (\beta_u \cdot (\boldsymbol{b}_u^{\mathsf{mul}}[v_i^*] \oplus \boldsymbol{b}_{v_i^*}^{\mathsf{mul}}[u])) \oplus (\boldsymbol{b}_u^{\mathsf{add}}[v_i^*] \oplus \boldsymbol{b}_{v_i^*}^{\mathsf{add}}[u]),$$
where β_u, $\boldsymbol{b}_u^{\mathsf{mul}}[v_i^*]$, $\boldsymbol{b}_{v_i^*}^{\mathsf{mul}}[u]$, $\boldsymbol{b}_u^{\mathsf{add}}[v_i^*]$ and $\boldsymbol{b}_{v_i^*}^{\mathsf{add}}[u]$ are values the simulator sampled in the previous steps. For every $v \in [2n+1] \setminus \{R, v_i^*, u\}$ the simulator samples at random $\boldsymbol{s}_u[v] \leftarrow \mathbb{F}_q$. Otherwise, the only corrupted neighbor u is the receiver, and for every $v \in [2n+1] \setminus \{R, u\}$ the simulator samples at random $\boldsymbol{s}_u[v] \leftarrow \mathbb{F}_q$. Eventually the simulator sends all the vectors $\boldsymbol{s}_u$ to the receiver on behalf of each u.

Having defined the simulator, we proceed to show that the view of the adversary in the simulated protocol is identically distributed as its view in a real execution of the protocol. Similarly to the previous proofs the simulation mirrors the protocol except for the last step, which is the only one that needs to be analyzed.

We proceed by analyzing various corruption patterns. We emphasize that the order of the cases matters, since when entering a case we assume that none of the prior cases holds. We begin by listing the "base" cases for one corruption and two corruptions; more corruptions will be reduced to these "base" cases.

Single corrupted party:

Case 1.1: Isolated corrupted party. No messages are received or sent, just as in the real execution; therefore, the view is identically distributed.

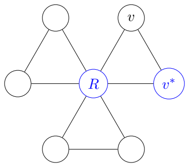
Case 1.2: Non-isolated corrupted party. The proof follows identically as the proof of Lemma 3.6.

Two corrupted parties:

Case 2.1: At least one corrupted party is isolated. The isolated party does not receive or send any messages, and the second party's view reduces to Case 1.

Case 2.2: Honest receiver. At Step 4 no messages are received. In the previous steps of the protocol the simulation acts the same as the real execution; therefore, the view is identically distributed.

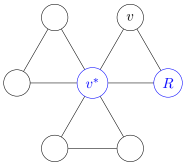
Case 2.3: Corrupted receiver is center. Let v^* and R be the corrupted parties, where R is the receiver. Denote by v the honest common neighbor of the corrupted parties; i.e., $v \in \mathcal{N}(v^*) \cap \mathcal{N}(R)$. At Step 4 the receiver receives messages from every neighbor, including v.



For every $u \in \mathcal{N}(R)$, the receiver receives a vector $\boldsymbol{s}_u$ in both the real and the simulated executions.

- When $u = v^*$, both P_u and the receiver are corrupted, and no messages are sent from the simulator.
- When $u = v$, in both the simulation and in the real execution the value $\boldsymbol{s}_v[v^*]$ is computed as
$$\boldsymbol{s}_v[v^*] = (\beta_v^{\text{2-nbr}} \cdot (\boldsymbol{b}_v^{\mathsf{mul}}[v^*] \oplus \boldsymbol{b}_{v^*}^{\mathsf{mul}}[v])) \oplus (\boldsymbol{b}_v^{\mathsf{add}}[v^*] \oplus \boldsymbol{b}_{v^*}^{\mathsf{add}}[v]),$$
where in the simulation, the values $\beta_v^{\text{2-nbr}}$, $\boldsymbol{b}_v^{\mathsf{mul}}[v^*]$, $\boldsymbol{b}_{v^*}^{\mathsf{mul}}[v]$, $\boldsymbol{b}_v^{\mathsf{add}}[v^*]$ and $\boldsymbol{b}_{v^*}^{\mathsf{add}}[v]$ were sampled in the previous steps and are known to Sim. Since v has two neighbors, the receiver and v^*, for every $w \in [2n+1] \setminus \{R, v^*, v\}$ the value $\boldsymbol{s}_v[w]$ are sampled uniformly at random both in the simulation and in the real execution, and $\boldsymbol{s}_v[R]$ and $\boldsymbol{s}_v[v]$ are zero both the simulation and real execution. Hence, the views in simulation and in the real execution are identically distributed.
- When $u \notin \{v, v^*\}$, Sim initializes a vector $\boldsymbol{s}_u$ of size $2n+1$ to zeros, and for every $w \in [2n+1] \setminus \{R, u\}$ samples uniformly at random $\boldsymbol{s}_u[w] \leftarrow \mathbb{F}_q$; in the real execution, if $w \notin \mathcal{N}(u)$ then $\boldsymbol{s}_u[w]$ is indeed sampled uniformly at random, but otherwise, it is computed as
$$\begin{aligned}\boldsymbol{s}_u[w] &= (\beta_u^{\text{2-nbr}} \cdot (\boldsymbol{b}_u^{\mathsf{mul}}[w] \oplus \boldsymbol{b}_w^{\mathsf{mul}}[u])) \oplus (\boldsymbol{b}_u^{\mathsf{add}}[w] \oplus \boldsymbol{b}_w^{\mathsf{add}}[u]) \oplus m_u \\ &= (\beta_u^{\text{2-nbr}} \cdot (\boldsymbol{b}_u^{\mathsf{mul}}[w] \oplus \boldsymbol{b}_w^{\mathsf{mul}}[u])) \oplus (\boldsymbol{b}_u^{\mathsf{add}}[w] \oplus \boldsymbol{b}_w^{\mathsf{add}}[u]).\end{aligned}$$
This equality holds since P_R is the center, so each of its neighbors sets $m_u = 0$. Thus, it is left to show that when w is a neighbor of u, the value $\boldsymbol{s}_u[w]$ is distributed uniformly at random from the view of the receiver (as done in the simulation).
Indeed, in the view of P_R, the values $\boldsymbol{s}_u[w]$ and $\boldsymbol{s}_w[u]$ are the only values that are based on $\boldsymbol{b}_u^{\mathsf{mul}}[w]$, $\boldsymbol{b}_w^{\mathsf{mul}}[u]$, $\boldsymbol{b}_u^{\mathsf{add}}[w]$, and $\boldsymbol{b}_w^{\mathsf{add}}[u]$, which in turn are each sampled independently and uniformly at random, and remain unknown to P_R. In addition, note that the values $\beta_u^{\text{2-nbr}}$ and $\beta_w^{\text{2-nbr}}$ are *known* to P_R (since P_R sampled and sent these values), and by Step 3 we are guaranteed that $\beta_u^{\text{2-nbr}} \neq \beta_w^{\text{2-nbr}}$ because P_R is the center. Denote,
$$b^{\mathsf{mul}} = \boldsymbol{b}_u^{\mathsf{mul}}[w] \oplus \boldsymbol{b}_w^{\mathsf{mul}}[u] \quad \text{and} \quad b^{\mathsf{add}} = \boldsymbol{b}_w^{\mathsf{add}}[u] \oplus \boldsymbol{b}_u^{\mathsf{add}}[w].$$
Note that b^{mul} and b^{add} are uniformly distributed. Looking back at $\boldsymbol{s}_u[w]$ and $\boldsymbol{s}_w[u]$, it holds that
$$\boldsymbol{s}_u[w] = (\beta_u^{\text{2-nbr}} \cdot b^{\mathsf{mul}}) \oplus b^{\mathsf{add}} \quad \text{and} \quad \boldsymbol{s}_w[u] = (\beta_w^{\text{2-nbr}} \cdot b^{\mathsf{mul}}) \oplus b^{\mathsf{add}}.$$
From the view of P_R, these are two linear equations with two unknowns b^{mul} and b^{add}, which are uniformly distributed. These equations are solvable since $\beta_u^{\text{2-nbr}} \neq \beta_w^{\text{2-nbr}}$. Therefore, from the view of P_R, the values $\boldsymbol{s}_u[w]$ and $\boldsymbol{s}_w[u]$ are uniformly distributed.

Case 2.4: Corrupted center and receiver. Since the receiver is not the center, it has two neighbors: the corrupted center v^* and an honest neighbor denoted by v. At Step 4 the receiver gets messages from v^* and from v.



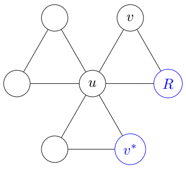
For both the simulation and the real world, the message from v is the vector $\boldsymbol{s}_u$ where $\boldsymbol{s}_v[v^*]$ is computed as

$$\boldsymbol{s}_v[v^*] \coloneqq (\beta_v \cdot (\boldsymbol{b}_v^{\mathsf{mul}}[v^*] \oplus \boldsymbol{b}_{v^*}^{\mathsf{mul}}[v])) \oplus (\boldsymbol{b}_v^{\mathsf{add}}[v^*] \oplus \boldsymbol{b}_{v^*}^{\mathsf{add}}[v]) \oplus m_u.$$

In the simulation, the values β_u, $\boldsymbol{b}_u^{\mathsf{mul}}[v]$, $\boldsymbol{b}_v^{\mathsf{mul}}[u]$, $\boldsymbol{b}_u^{\mathsf{add}}[v]$ and $\boldsymbol{b}_v^{\mathsf{add}}[u]$ are known from the previous steps and the value m_u is set to 0 since v is not the center; in the real world, the value $\boldsymbol{s}_v[v^*]$ is formed in the same way. The values $\boldsymbol{s}_v[v]$ and $\boldsymbol{s}_v[R]$ are set to zero both in the simulation and in the real execution, and for every $u \notin \{R, v, v^*\}$, the value $\boldsymbol{s}_v[u]$ are sampled uniformly at random.

Case 2.5: Corrupted receiver is non-center, center is honest. In this case, the receiver has two neighbors.

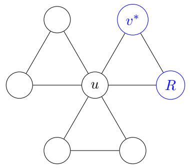
- If both are honest, then at Step 4 the receiver receives messages from both neighbors. By the structure of friendship graphs, the common neighbor of the corrupted parties, denoted by u, is the center; denote by v the second neighbor of the receiver. In addition, denote the non-receiver corrupted party v^*.



  We will focus on the $\boldsymbol{s}_v$ and $\boldsymbol{s}_u$ received by P_R from P_v and P_u, respectively. Note that $\boldsymbol{s}_v[R] = \boldsymbol{s}_u[R] = 0$, that $\boldsymbol{s}_v[v] = 0$ and that $\boldsymbol{s}_u[u] = 0$, both in the simulation and in the real execution. Further, the values $\boldsymbol{s}_v[u]$ and $\boldsymbol{s}_u[v]$, are sampled in the simulation conditioned on $\boldsymbol{s}_v[u] \oplus \boldsymbol{s}_u[v] = m$, which is identically distributed as the real execution. In addition, the value sent by P_u about the non-receiver corrupted party P_{v^*} is formed as

$$\boldsymbol{s}_u[v^*] \coloneqq (\beta_u \cdot (\boldsymbol{b}_u^{\mathsf{mul}}[v^*] \oplus \boldsymbol{b}_{v^*}^{\mathsf{mul}}[u])) \oplus (\boldsymbol{b}_u^{\mathsf{add}}[v^*] \oplus \boldsymbol{b}_{v^*}^{\mathsf{add}}[u]) \oplus m_u,$$

  where the values β_u, $\boldsymbol{b}_u^{\mathsf{mul}}[v^*]$, $\boldsymbol{b}_{v^*}^{\mathsf{mul}}[u]$, $\boldsymbol{b}_u^{\mathsf{add}}[v^*]$, $\boldsymbol{b}_{v^*}^{\mathsf{add}}[u]$, and m_u were evaluated in the previous steps of the protocol, and used accordingly both in the simulation and in the real execution; hence, the views are identically distributed.
  Finally, the values $\boldsymbol{s}_v[w]$ for every $w \in [2n+1] \setminus \{R, v, u\}$ and $\boldsymbol{s}_u[w]$ for every $w \in [2n+1] \setminus \{R, v, u, v^*\}$ are sampled uniformly at random.
  To conclude, the vectors $\boldsymbol{s}_v$ and $\boldsymbol{s}_u$ are identically distributed as the real execution.
- Otherwise, if not both neighbors are honest, then one of the receiver's neighbors is corrupted, and the other is the center. Denote the center by u, and the non-center corrupted party by v^*.



The receiver gets messages from both neighbors. Specifically, P_R gets the vector $\boldsymbol{s}_u$ from the center, where $\boldsymbol{s}_u[R] = \boldsymbol{s}_u[u] = 0$, for $w \in [2n+1] \setminus \{R, u, v^*\}$ the value $\boldsymbol{s}_u[w]$ is sampled uniformly at random, and the value $\boldsymbol{s}_u[v^*]$ that is formed as

$$\boldsymbol{s}_u[v^*] := (\beta_u \cdot (\boldsymbol{b}_u^{\mathsf{mul}}[v^*] \oplus \boldsymbol{b}_{v^*}^{\mathsf{mul}}[u])) \oplus (\boldsymbol{b}_u^{\mathsf{add}}[v^*] \oplus \boldsymbol{b}_{v^*}^{\mathsf{add}}[u]) \oplus m_u$$

based on values β_u, $\boldsymbol{b}_u^{\mathsf{mul}}[v^*]$, $\boldsymbol{b}_{v^*}^{\mathsf{mul}}[u]$, $\boldsymbol{b}_u^{\mathsf{add}}[v^*]$, and $\boldsymbol{b}_{v^*}^{\mathsf{add}}[u]$ that were sent in the previous steps of the protocol. In addition, $m_u = m$ which is known to Sim. Hence, the views in the simulation and in the real execution are identically distributed.

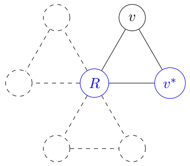
More than 2 corrupted parties:

Case 3.1: At most two corrupted parties are not isolated. In this case, the isolated corrupted parties do not receive or send any messages, so their view is identical to the real execution. In addition, the non-isolated corrupted parties' view reduces immediately to the previous cases.

Case 3.1: Honest receiver. At Step 4 no messages are received. In the previous steps of the protocol the simulation acts the same as the real execution; therefore, the views are identically distributed.

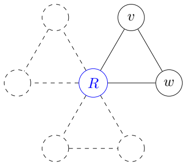
Case 3.2: Corrupted receiver is center. By the structure of the protocol, we can decompose the view of the adversary to its view in every triangle which are independent of each other. Note that the corrupted center appears in every triangle of the friendship graph; therefore, in every triangle there are 0, 1, or 2 honest parties.

- If the triangle is fully corrupted, then simulating the messages in the triangle is immediate.
- If there is exactly one honest party in the triangle, denote by v the honest party and by v^* the non-center corrupted party.



In Step 4, P_v sends $\boldsymbol{s}_v$ to the receiver. Note that in both the simulation and the real execution, the values in the vector $\boldsymbol{s}_v$ are set as follows: $\boldsymbol{s}_v[v] = \boldsymbol{s}_v[R] = 0$, for every $u \in [2n+1] \setminus \{R, v, v^*\}$ the value $\boldsymbol{s}_v[u]$ is sampled uniformly at random, and for $u = v^*$

$$\boldsymbol{s}_v[u] := (\beta_v^{\text{2-nbr}} \cdot (\boldsymbol{b}_v^{\mathsf{mul}}[u] \oplus \boldsymbol{b}_u^{\mathsf{mul}}[v])) \oplus (\boldsymbol{b}_v^{\mathsf{add}}[u] \oplus \boldsymbol{b}_u^{\mathsf{add}}[v]),$$



where the values $\beta_v^{\text{2-nbr}}$, $\boldsymbol{b}_v^{\mathsf{mul}}[u]$, $\boldsymbol{b}_u^{\mathsf{mul}}[v]$, $\boldsymbol{b}_v^{\mathsf{add}}[u]$, and $\boldsymbol{b}_u^{\mathsf{add}}[v]$ were sent in the previous steps of the protocol, in both the simulation and in the real execution, and therefore are identically distributed.

- If there are two honest parties, denote them by v and w.
  At Step 4, v and w send the receiver the vectors $\boldsymbol{s}_v$ and $\boldsymbol{s}_w$, respectively. Note that $\boldsymbol{s}_u[R] = \boldsymbol{s}_w[R] = 0$, that $\boldsymbol{s}_v[v] = 0$ and that $\boldsymbol{s}_w[w] = 0$. For $u \in [2n+1] \setminus \{R, v^*, v\}$ the value $\boldsymbol{s}_v[u]$ is sampled uniformly at random, and for $u \in [2n+1] \setminus \{R, v^*, w\}$ the value $\boldsymbol{s}_w[u]$ is sampled uniformly at random. The values $\boldsymbol{s}_w[v]$ and $\boldsymbol{s}_v[w]$ in the real execution are formed as follows:

  $$\boldsymbol{s}_v[w] := (\beta_v^{\text{2-nbr}} \cdot (\boldsymbol{b}_v^{\mathsf{mul}}[w] \oplus \boldsymbol{b}_w^{\mathsf{mul}}[v])) \oplus (\boldsymbol{b}_v^{\mathsf{add}}[w] \oplus \boldsymbol{b}_w^{\mathsf{add}}[v])$$
  $$\boldsymbol{s}_w[v] := (\beta_w^{\text{2-nbr}} \cdot (\boldsymbol{b}_w^{\mathsf{mul}}[v] \oplus \boldsymbol{b}_v^{\mathsf{mul}}[w])) \oplus (\boldsymbol{b}_w^{\mathsf{add}}[v] \oplus \boldsymbol{b}_v^{\mathsf{add}}[w]),$$

  where in the simulation, both $\boldsymbol{s}_v[w]$ and $\boldsymbol{s}_w[v]$ are sampled uniformly at random. It is left to show that those values are distributed uniformly at random from the view of the receiver. Since the receiver is the center, it is guaranteed that $\beta_v^{\text{2-nbr}} \neq \beta_w^{\text{2-nbr}}$ and are known to P_R. The values $\boldsymbol{b}_v^{\mathsf{mul}}[w]$, $\boldsymbol{b}_w^{\mathsf{mul}}[v]$, $\boldsymbol{b}_v^{\mathsf{add}}[w]$, and $\boldsymbol{b}_w^{\mathsf{add}}[v]$, which were sampled uniformly at random, are used in the receiver's view only when computing $\boldsymbol{s}_v[w]$ and $\boldsymbol{s}_w[v]$, and remain unknown to P_R. Denote,

  $$b^{\mathsf{mul}} = \boldsymbol{b}_u^{\mathsf{mul}}[w] \oplus \boldsymbol{b}_w^{\mathsf{mul}}[u] \quad \text{and} \quad b^{\mathsf{add}} = \boldsymbol{b}_w^{\mathsf{add}}[u] \oplus \boldsymbol{b}_u^{\mathsf{add}}[w].$$

  Note that b^{mul} and b^{add} are uniformly distributed. Looking back at $\boldsymbol{s}_u[w]$ and $\boldsymbol{s}_w[u]$, it holds that

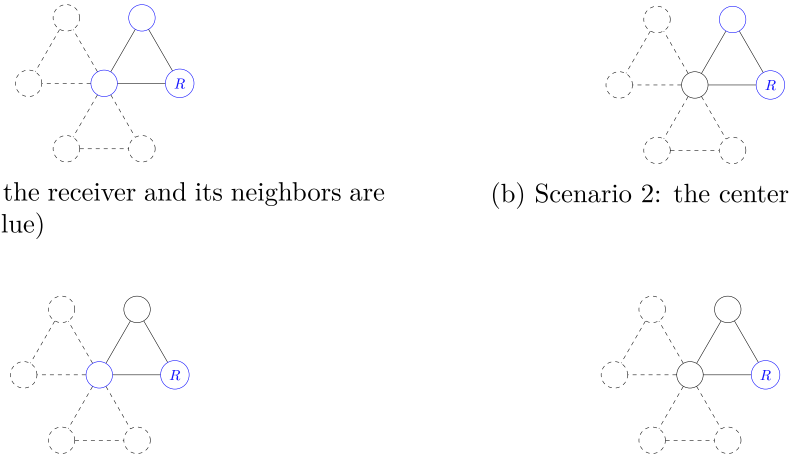
  $$\boldsymbol{s}_u[w] = (\beta_u^{\text{2-nbr}} \cdot b^{\mathsf{mul}}) \oplus b^{\mathsf{add}} \quad \text{and} \quad \boldsymbol{s}_w[u] = (\beta_w^{\text{2-nbr}} \cdot b^{\mathsf{mul}}) \oplus b^{\mathsf{add}}.$$

  From the view of P_R, these are two linear equations with two unknowns b^{mul} and b^{add}, which are uniformly distributed. These equations are solvable since $\beta_u^{\text{2-nbr}} \neq \beta_w^{\text{2-nbr}}$. Therefore, the simulated view and the real execution are identically distributed.

Case 3.4: Corrupted receiver is not center. Until Step 4, both the simulation and the real execution follow the protocol, and during this step, only the receiver (which is non-center) receives messages from its two neighbors. The possible scenarios are as follows:

- If both of the receiver's neighbors are corrupted, the simulation is immediate. See Scenario 1 below.
- If there is only one honest neighbor who is the center, then for each corrupted neighbor of the center the simulator acts exactly as the real execution, and for honest neighbors the case reduces to Case 2.5; hence, the view is identically distributed. See Scenario 2 below.

- If the center (which is non-receiver) is corrupted, the simulation runs exactly the same as the real execution; hence, the view is identically distributed. See Scenario 3 below.
- If both of the receiver's neighbors are honest, then using induction on the center's corrupted neighbors and using Case 2.5 in each step to show that the view is identically distributed. See Scenario 4 below.



(a) Scenario 1: the receiver and its neighbors are corrupted (in blue)

(b) Scenario 2: the center is honest

(c) Scenario 3: the non-center neighbor of the receiver is honest

(d) Scenario 4: both of the receiver's neighbors are honest

This concludes the proof of Theorem 3.7. □

4 1-IT-THB Admissible Graph Class

In this section, we present our 1-IT-THB protocol for admissible graph classes. We begin, in Section 4.1, with the special case of wheel graphs; next, in Section 4.2, we present the protocol for arbitrary admissible graphs.

4.1 1-IT-THB for Wheel Graphs

We begin by constructing 1-IT-THB for the wheel graph-class. First, we formally define the graph class as the isomorphically closed class of W_n (see Definition 2.1).

Definition 4.1 (wheel graph). *Let $n \in \mathbb{N}$ such that $n \geq 4$. The wheel graph W_n consists of $n+1$ nodes: a cycle of size n and a center node v connected to all other nodes.*

Definition 4.2 (wheel graph-class). *Let $n \in \mathbb{N}$ such that $n \geq 4$. The* wheel graph-class $\mathcal{G}_{\mathsf{wheel}}(n)$ *is the isomorphically closed graph class associated with W_n.*

Notations used in the protocol. We refer the reader to Section 1.2 for a high-level overview. Below, we describe the notation used in the construction.

- The matrix of *correlated values*, $\mathbf{C}_u^{\text{3-nbr}} \in \mathbb{F}_q^{(n+1)\times(n+1)}$, is sampled uniformly at random by every non-receiver party P_u. Some values from these matrices are exchanged between neighboring parties, and are used as secret-shares of the sender's message within the matrices $\mathbf{M}_u$ that are sent to the receiver.

We note that not all values in these matrices will actually be used in the protocol; we chose to use matrices with redundant random values for simplicity of notation. Further, for consistency with the protocol for admissible graphs (Section 4.2), we emphasize in the notation of $\mathbf{C}_u^{\text{3-nbr}}$ that these matrices are only relevant when the receiver has 3 neighbors.

- The matrix $\mathbf{M}_u \in \mathbb{F}_q^{(n+1)\times(n+1)}$ is generated by every party P_u that is a neighbor of the receiver, and is sent by P_u to the receiver in the last step of the protocol. Some of the values in this matrix are uniformly sampled, and if the receiver has three neighbors (one of which is the center) the corresponding entries in these matrices will be crafted as 3-out-of-3 shares of the message (using the correlated values).

Similarly to Section 3.1, we first analyze the protocol π_{wheel} (formally defined in Figure 17) for fixed-size wheel graphs, where there is a single connected component. However, we add the instructions (in blue) for handling variable-size graphs, in which there is one main connected component (that is a wheel) and all other nodes are isolated (see Corollary 4.6).

Theorem 4.3 (IT-THB for wheel graph-class)**.** *Let $n \in \mathbb{N}$ such that $n \geq 4$ and consider the wheel graph-class $\mathcal{G}_{\mathsf{wheel}}(n)$. The protocol π_{wheel} (defined in Figure 17) is a perfectly secure* IT-THB *protocol against a single semi-honest corruption with respect to $\mathcal{G}_{\mathsf{wheel}}(n)$.*

Proof. First, note that the protocol is well-defined; because $|V| = n+1 \geq 5$, there exists a unique center node $v \in V$ with degree $\deg(v) = |V|-1$. We proceed to prove *correctness* and *security.*

Correctness. In case P_R is a neighbor of P_S, then in the first round P_R receives m and sets m as its output. In case P_R is *not* a neighbor of P_S, then P_R has 3 neighbors. Denote $\mathcal{N}(R) = \{v_1, v_2, u\}$; by the structure of wheel graphs, one of these neighbors, say P_u, is the center. The output in this case is defined to be

$$\begin{aligned}
&\mathbf{M}_{v_1}[v_2,u] \oplus \mathbf{M}_{v_2}[v_1,u] \oplus \mathbf{M}_u[v_1,v_2] = \\
&= \big(\sigma^u_{v_2,v_1} \oplus \sigma^{v_2}_{u,v_1} \oplus m_{v_1}\big) \oplus \big(\sigma^u_{v_1,v_2} \oplus \sigma^{v_1}_{u,v_2} \oplus m_{v_2}\big) \oplus \big(\sigma^{v_2}_{v_1,u} \oplus \sigma^{v_1}_{v_2,u} \oplus m_u\big) \\
&= \big((0 \oplus \sigma^{v_2}_{u,v_1}) \oplus m_{v_1}\big) \oplus \big((0 \oplus \sigma^{v_1}_{u,v_2}) \oplus m_{v_2}\big) \oplus \big((\sigma^{v_2}_{v_1,u} \oplus \sigma^{v_1}_{v_2,u}) \oplus m_u\big) \\
&= \big((0 \oplus \mathbf{C}_u^{\text{3-nbr}}[v_1,v_2]) \oplus m_{v_1}\big) \oplus \big((0 \oplus \mathbf{C}_u^{\text{3-nbr}}[v_2,v_1]) \oplus m_{v_2}\big) \oplus \big((\mathbf{C}_u^{\text{3-nbr}}[v_1,v_2] \oplus \mathbf{C}_u^{\text{3-nbr}}[v_2,v_1]) \oplus m_u\big) \\
&= (\mathbf{C}_u^{\text{3-nbr}}[v_1,v_2] \oplus \mathbf{C}_u^{\text{3-nbr}}[v_2,v_1] \oplus \mathbf{C}_u^{\text{3-nbr}}[v_1,v_2] \oplus \mathbf{C}_u^{\text{3-nbr}}[v_2,v_1]) \oplus (m_{v_1} \oplus m_{v_2} \oplus m_u) \\
&= (0 \oplus 0) \oplus (0 \oplus 0 \oplus m) = m.
\end{aligned}$$

The equalities are justified as follows:

- The first equality holds by the way $\mathbf{M}_{v_1}[v_2,u]$, $\mathbf{M}_{v_2}[v_1,u]$, and $\mathbf{M}_u[v_1,v_2]$ are defined in Step 3.
- The second equality holds by Step 3: since the center u is the common neighbor of v_1 and v_2, which are not neighbors (by the structure of wheel graphs), so $\sigma^u_{v_1,v_2}$ and $\sigma^u_{v_2,v_1}$ are zero.
- The third equality holds by Step 3: since u is the center the value $\sigma^{v_{3-i}}_{v_i,u} = \mathbf{C}_u^{\text{3-nbr}}[v_i, v_{3-i}]$ for $i \in \{1,2\}$ and since v_1 and v_2 are not the center, it holds that $\sigma^{v_{3-i}}_{u,v_i} = \mathbf{C}_u^{\text{3-nbr}}[v_i, v_{3-i}]$.
- The last equality holds because non-center nodes set $m_{v_1} = m_{v_2} = 0$ in Step 3.

Protocol $\pi_{\mathsf{wheel}}(n, \mathsf{P}_S)$

Auxiliary input: A binary field $\mathbb{F}_q$ such that $q > n$.

Input: The sender P_S, with $S \in [n+1]$, holds an input $m \in \{0,1\}$.

Hybrid model: The protocol is defined in the $\mathcal{F}_{\mathsf{graph}}^{\mathcal{G}_{\mathsf{wheel}}(n)}$-hybrid model.

The protocol:

- Each party P_v sends an initialization message to $\mathcal{F}_{\mathsf{graph}}^{\mathcal{G}_{\mathsf{wheel}}(n)}$ and receives its neighbor-set $\mathcal{N}(v)$.
- Repeat for each potential receiver P_R with $R \in [n+1]$:

1. *Forwarding the message to the center node.* The sender P_S sends m to its neighbors.
   If P_S is a isolated, the center acts as if received $m = 0$ from P_S.
2. *Generating correlated values.* Every non-receiver party $\mathsf{P}_u \neq \mathsf{P}_R$ samples uniformly at random a matrix $\mathbf{C}_u^{\text{3-nbr}} \leftarrow \mathbb{F}_q^{(n+1)\times(n+1)}$. Next, P_u sends to every P_v with $v \in \mathcal{N}(u) \setminus \{R\}$ the row corresponding to v in $\mathbf{C}_u^{\text{3-nbr}}$.
   In case P_u is the center, it plays in its head every isolated P_v; that is, P_u samples on behalf of P_v a matrix $\mathbf{C}_v^{\text{3-nbr}} \leftarrow \mathbb{F}_q^{(n+1)\times(n+1)}$ and sends the row corresponding to u in $\mathbf{C}_v^{\text{3-nbr}}$ to itself.
3. *Forwarding m to P_R.* In this step, only parties P_u with $u \in \mathcal{N}(R)$ send messages, and only P_R receives messages. Party P_u proceeds as follows: Initialize a matrix $\mathbf{M}_u$ of size $(n+1)^2$ to zeros. For every $v_1, v_2 \in [n+1] \setminus \{R, u\}$ such that $v_1 \neq v_2$:
   (a) If $\{v_1, v_2\} \cap \mathcal{N}(u) = \emptyset$, sample a value $x \leftarrow \mathbb{F}_q$ at random and set both $\mathbf{M}_u[v_1, v_2] = x$ and $\mathbf{M}_u[v_2, v_1] = x$.
   (b) Otherwise, if $\{v_1, v_2\} \cap \mathcal{N}(u) \neq \emptyset$, set $\mathbf{M}_u[v_1, v_2] := \sigma^{v_2}_{v_1,u} \oplus \sigma^{v_1}_{v_2,u} \oplus m_u$, where
   – For $i \in \{1,2\}$: if u is the center set $\sigma^{v_{3-i}}_{v_i,u} := \mathbf{C}_u^{\text{3-nbr}}[v_i, v_{3-i}]$; otherwise, if $v_i \in \mathcal{N}(u)$ set $\sigma^{v_{3-i}}_{v_i,u} := \mathbf{C}_{v_i}^{\text{3-nbr}}[u, v_{3-i}]$; else $\sigma^{v_{3-i}}_{v_i,u} := 0$.
   – If u is the center node, set $m_u := m$; else $m_u := 0$.

   Finally, P_u sends to P_R the matrix $\mathbf{M}_u$.
4. P_R *output.*
   – If P_R is P_S then output its input.
   – If $R \in \mathcal{N}(S)$, output m as received on Step 1.
   – If $|\mathcal{N}(R)| = 3$, denote $\mathcal{N}(R) = \{v_1, v_2, v_3\}$ and output $\mathbf{M}_{v_1}[v_2, v_3] \oplus \mathbf{M}_{v_2}[v_1, v_3] \oplus \mathbf{M}_{v_3}[v_1, v_2]$.
   – If $|\mathcal{N}(R)| = 0$, output 0.

Figure 17: Information-theoretic 1-THB over $\mathcal{G}_{\mathsf{wheel}}(n)$. The instruction lines in blue are only needed when considering wheel graphs with variable size (i.e., when some nodes are isolated).

Security. We proceed to prove security. Let P_{v^*} with $v^* \in [n+1]$ denote the corrupted party. We will construct a simulator Sim that given the neighbor-set of P_{v^*} generates a simulated view for P_{v^*} that is identically distributed as its view in a real execution of the protocol. As we consider semi-honest security, and as broadcast is a deterministic functionality, this implies that the environment's output is identically distributed in the real and ideal computations.

The simulator Sim runs P_{v^*} in its head. Initially, P_{v^*} sends an initialization message to $\mathcal{F}_{\mathsf{graph}}^{\mathcal{G}}$; the simulator forwards this message to $\mathcal{W}_{\mathsf{graph\text{-}info}}^{\mathcal{G}}(\mathcal{F}_{\mathsf{bc}}(\mathsf{P}_S))$ and sends the response $\mathcal{N}(v^*)$ to P_{v^*}. Next, in case the sender is corrupted, i.e., $\mathsf{P}_{v^*} = \mathsf{P}_S$, the simulator sends its input m to

$\mathcal{W}^{\mathcal{G}}_{\textsf{graph-info}}(\mathcal{F}_{\textsf{bc}}(\mathsf{P}_S))$; regardless, Sim receives the message m as the output from $\mathcal{W}^{\mathcal{G}}_{\textsf{graph-info}}(\mathcal{F}_{\textsf{bc}}(\mathsf{P}_S))$ (formally, Sim sends the "empty input" for a non-sender corrupted party).

Next, Sim simulates an RMT instance towards the corrupted party, for every potential receiver P_R with $R \in [n+1] \setminus \{S\}$.

- To simulate Step 1, if the sender is corrupted, Sim receives m on behalf of every honest P_v for which $v \in \mathcal{N}(S)$. If the sender is honest and $S \in \mathcal{N}(v^*)$ then Sim sends m to P_{v^*}.
- To simulate Step 2, if P_{v^*} is not the receiver, for every honest neighbor P_u with $u \in \mathcal{N}(v^*)$, Sim samples uniformly at random from $\mathbb{F}_q^{(n+1)\times(n+1)}$ correlated values for a square matrix $\mathbf{C}_u^{\text{3-nbr}}$ and sends the corresponding row vector to P_{v^*} on behalf of P_u. In addition, Sim receives the vector $\mathbf{C}_{v^*}^{\text{3-nbr}}[u]$ from P_{v^*} on behalf of P_u.
- To simulate Step 3, if P_{v^*} is a neighbor of the receiver, i.e., $v^* \in \mathcal{N}(R)$, then Sim receives from P_{v^*} the matrix $\mathbf{M}_{v^*}$ on behalf of P_R.

  If P_{v^*} is the receiver, we consider two cases:

  1. If P_{v^*} is the center node, then for each $u \neq v^*$ the simulator first initializes a matrix $\mathbf{M}_u$ of size $(n+1)\times(n+1)$ to zeros; for every $v_1, v_2 \notin \{u, v^*\}$ such that $v_1 \neq v_2$, the simulator samples $x \leftarrow \mathbb{F}_q$ at random and sets $\mathbf{M}_u[v_1, v_2] = \mathbf{M}_u[v_2, v_1] = x$. Finally, the simulator sends $\mathbf{M}_u$ to P_{v^*} on behalf of each P_u.
  2. Otherwise, denote $\mathcal{N}(v^*) = \{w_1, w_2, w_3\}$. First the simulator initializes the matrices $\mathbf{M}_{w_1}$, $\mathbf{M}_{w_2}$, and $\mathbf{M}_{w_3}$, each of size $(n+1)\times(n+1)$ to zeros; next, the simulator randomly samples values $x_1, x_2, x_3 \leftarrow \mathbb{F}_q$ conditioned on $x_1 \oplus x_2 \oplus x_3 = m$ and sets

     $$\begin{aligned}\mathbf{M}_{w_1}[w_2, w_3] &= \mathbf{M}_{w_1}[w_3, w_2] = x_1\\ \mathbf{M}_{w_2}[w_1, w_3] &= \mathbf{M}_{w_2}[w_3, w_1] = x_2\\ \mathbf{M}_{w_3}[w_2, w_1] &= \mathbf{M}_{w_3}[w_1, w_2] = x_3.\end{aligned}$$

     For every $i \in \{1,2,3\}$ and every $v_1, v_2 \in [n+1] \setminus \{v^*, w_i\}$ such that $|\{v_1, v_2\} \cap \mathcal{N}(v^*)| < 2$ and $v_1 \neq v_2$, the simulator samples at random $x'_i \leftarrow \mathbb{F}_q$ and sets

     $$\mathbf{M}_{w_i}[v_1, v_2] = \mathbf{M}_{w_i}[v_2, v_1] = x'_i.$$

     Finally, the simulator sends to the adversary the matrices $\mathbf{M}_{w_1}$, $\mathbf{M}_{w_2}$, and $\mathbf{M}_{w_3}$.

We proceed to show that the view of P_{v^*} in the simulated protocol is identically distributed as its view in a real execution of the protocol. Note that the simulation mirrors the protocol except for the last step (Step 3). Therefore, we need to analyze only the last step where v^* is the receiver.

Case 1: P_{v^*} has three neighbors. That is, P_{v^*} is not the center; denote $\mathcal{N}(v^*) = \{w_1, w_2, w_3\}$. Since one of the neighbors is the center, in the real execution it holds that

$$\mathbf{M}_{w_1}[w_2, w_3] \oplus \mathbf{M}_{w_2}[w_1, w_3] \oplus \mathbf{M}_{w_3}[w_1, w_2] = m.$$

Indeed, the simulation guarantees that all values are uniformly sampled from $\mathbb{F}_q$ conditioned on $\mathbf{M}_{w_1}[w_2, w_3] \oplus \mathbf{M}_{w_2}[w_1, w_3] \oplus \mathbf{M}_{w_3}[w_1, w_2] = m$; hence, these triplets are identically distributed in the real and simulated executions.

For every $v_1, v_2 \in [n+1]$ and every $i \in \{1,2,3\}$, the values of the matrix $\mathbf{M}_{w_i}[v_1, v_2]$ are set as follows during the execution in the real world:

1. If $\{v_1, v_2\} \cap \{R, w_i\} \neq \emptyset$ or if $v_1 = v_2$, it holds that $\mathbf{M}_{w_i}[v_1, v_2] = 0$; as in the simulation.
2. If $\{v_1, v_2\} \cap \mathcal{N}(w_i) = \emptyset$, then $\mathbf{M}_{w_i}[v_1, v_2]$ is sampled uniformly at random; as in the simulation.
3. If w_i is not the center and $\{v_1, v_2\} \cap \mathcal{N}(w_i) = \{v_j\}$ with $j \in \{1, 2\}$ (i.e., $|\{v_1, v_2\} \cap \mathcal{N}(w_i)| = 1$), then, according to Step 3, $\mathbf{M}_{w_i}[v_{3-j}, v_j]$ is set as:

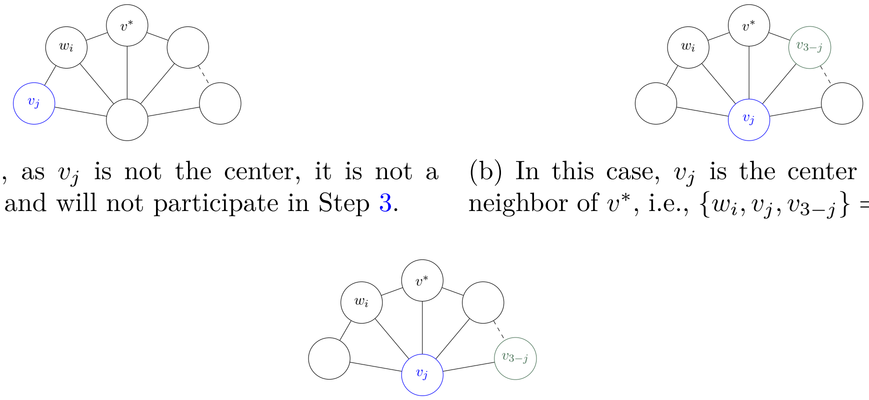
$$\begin{aligned}\mathbf{M}_{w_i}[v_{3-j}, v_j] &= \sigma^{v_j}_{v_{3-j},w_i} \oplus \sigma^{v_{3-j}}_{v_j,w_i} \oplus m_{w_i}\\ &= \sigma^{v_j}_{v_{3-j},w_i} \oplus \sigma^{v_{3-j}}_{v_j,w_i} \oplus 0\\ &= 0 \oplus \mathbf{C}^{\text{3-nbr}}_{v_j}[w_i, v_{3-j}],\end{aligned}$$

where the second equality holds since a non-center w_i sets $m_{w_i} = 0$, and the third equality holds since a non-center w_i sets $\sigma^{v_{3-j}}_{v_j,w_i} = \mathbf{C}^{\text{3-nbr}}_{v_j}[w_i, v_{3-j}]$ for $v_j \in \mathcal{N}(w_i)$, and sets $\sigma^{v_j}_{v_{3-j},w_i} = 0$ for $v_{3-j} \notin \mathcal{N}(w_i)$.

The only other party that can use the value $\mathbf{C}^{\text{3-nbr}}_{v_j}[w_i, v_{3-j}]$ is the center.[7] If v_j is the center (as illustrated in Figures 18b and 18c), the relevant value $\mathbf{M}_{v_j}[w_i, v_{3-j}]$ is formed as:

$$\begin{aligned}\mathbf{M}_{v_j}[w_i, v_{3-j}] &= \sigma^{v_{3-j}}_{w_i,v_j} \oplus \sigma^{w_i}_{v_{3-j},v_j} \oplus m_{v_j}\\ &= \sigma^{v_{3-j}}_{w_i,v_j} \oplus \sigma^{w_i}_{v_{3-j},v_j} \oplus m\\ &= \mathbf{C}^{\text{3-nbr}}_{v_j}[w_i, v_{3-j}] \oplus \mathbf{C}^{\text{3-nbr}}_{v_j}[v_{3-j}, w_i] \oplus m.\end{aligned}$$

The value $\mathbf{C}^{\text{3-nbr}}_{v_j}[v_{3-j}, w_i]$ is sent from the center v_j to v_{3-j}. If v_{3-j} is a neighbor of the receiver (Figure 18b), then $\{w_i, v_j, v_{3-j}\} = \{w_1, w_2, w_3\}$; else, if v_{3-j} is not a neighbor of the receiver (Figure 18c), then the value $\mathbf{C}^{\text{3-nbr}}_{v_j}[v_{3-j}, w_i]$ is not used in any other message that is part of the receiver's view and is identically distributed as the simulated value which is sampled uniformly at random.



(a) In this case, as v_j is not the center, it is not a neighbor of P_{v^*} and will not participate in Step 3.

(b) In this case, v_j is the center and v_{3-j} is a neighbor of v^*, i.e., $\{w_i, v_j, v_{3-j}\} = \{w_1, w_2, w_3\}$

(c) In this case, v_j is the center and v_{3-j} is not a neighbor of v^*; hence, the value $\mathbf{C}^{\text{3-nbr}}_{v_j}[v_{3-j}, w_i]$ will not be used.

Figure 18: The various settings in Case 1, Item 3, where the corrupted receiver P_{v^*} is not the center and $\{v_1, v_2\} \cap \mathcal{N}(w_i) = \{v_j\}$ for one of P_{v^*}'s neighbors w_i regarding arbitrary parties v_1 and v_2. In particular, w_i is not the center and it may use the correlated value $\mathbf{C}^{\text{3-nbr}}_{v_j}[w_i, v_{3-j}]$ sent by v_j, and known only to w_i and v_j.

[7] We highlight some values with blue and green colors to match the illustration in Figure 18.

4. If w_i is not the center and $\{v_1, v_2\} \subseteq \mathcal{N}(w_i)$ (illustrated in Figure 19) then, according to Step 3, $\mathbf{M}_{w_i}[v_{3-j}, v_j]$ is formed by:

$$\begin{aligned}\mathbf{M}_{w_i}[v_1, v_2] &= \sigma^{v_2}_{v_1,w_i} \oplus \sigma^{v_1}_{v_2,w_i} \oplus m_{w_i} \\ &= \sigma^{v_2}_{v_1,w_i} \oplus \sigma^{v_1}_{v_2,w_i} \oplus 0 \\ &= \mathbf{C}^{\text{3-nbr}}_{v_1}[w_i, v_2] \oplus \mathbf{C}^{\text{3-nbr}}_{v_2}[w_i, v_1] \oplus 0,\end{aligned}$$

where the second equality holds since the non-center w_i sets $m_{w_i} = 0$, and the third equality holds according to Step 3, as $\sigma^{v_2}_{v_1,w_i} = \mathbf{C}^{\text{3-nbr}}_{v_1}[w_i, v_2]$ and $\sigma^{v_1}_{v_2,w_i} = \mathbf{C}^{\text{3-nbr}}_{v_2}[w_i, v_1]$.

By the structure of the wheel graph, as w_i is not the center, either v_1 or v_2 is not a neighbor of the receiver P_{v^*}; assume without loss of generality that v_1 is not a neighbor of P_{v^*} and consider the value $\mathbf{C}^{\text{3-nbr}}_{v_1}[w_i, v_2]$: the only time when this value is communicated is when party v_1 sends it to party w_i at Step 2. During the simulation, Sim samples $\mathbf{M}_{w_i}[v_1, v_2]$ uniformly at random and independently of all other values; hence, $\mathbf{M}_{w_i}[v_1, v_2]$ is identically distributed in the real execution and the simulated one.

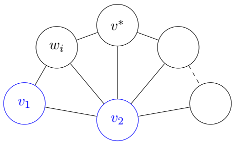


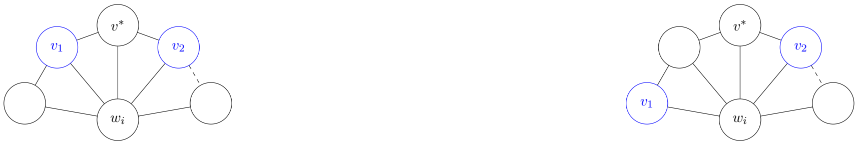
Figure 19: The setting in Case 1, Item 4, where the corrupted receiver P_{v^*} is not the center and v_1 and v_2 are the non-receiver neighbors of w_i (colored in blue).

5. If w_i is the center, then in particular $\{v_1, v_2\} \subseteq \mathcal{N}(w_i)$ (see Figure 20). According to Step 3, $\mathbf{M}_{w_i}[v_1, v_2]$ is formed as:

$$\begin{aligned}\mathbf{M}_{w_i}[v_1, v_2] &= \sigma^{v_2}_{v_1,w_i} \oplus \sigma^{v_1}_{v_2,w_i} \oplus m_{w_i} \\ &= \sigma^{v_2}_{v_1,w_i} \oplus \sigma^{v_1}_{v_2,w_i} \oplus m \\ &= \mathbf{C}^{\text{3-nbr}}_{w_i}[v_1, v_2] \oplus \mathbf{C}^{\text{3-nbr}}_{w_i}[v_2, v_1] \oplus m,\end{aligned}$$

where the second equality holds since the center w_i sets $m_{w_i} = m$, and the third equality holds since w_i is the center so $\sigma^{v_2}_{v_1,w_i} = \mathbf{C}^{\text{3-nbr}}_{w_i}[v_1, v_2]$ and $\sigma^{v_1}_{v_2,w_i} = \mathbf{C}^{\text{3-nbr}}_{w_i}[v_2, v_1]$.

Note that the case where $\{v_1, v_2, w_i\} \subseteq \{w_1, w_2, w_3\}$ (Figure 20a) is handled above, at the beginning of Case 1. Otherwise, without loss of generality, let $v_1 \notin \{w_1, w_2, w_3\}$ (Figure 20b); in this case, the value $\mathbf{C}^{\text{3-nbr}}_{w_i}[v_1, v_2]$ is sent from w_i to v_1 in Step 2, and is only used in $\mathbf{M}_{w_i}[v_1, v_2]$ (since v_1 is not a neighbor of the receiver). During the simulation, Sim samples $\mathbf{M}_{w_i}[v_1, v_2]$ uniformly at random and independently of all other values; hence, $\mathbf{M}_{w_i}[v_1, v_2]$ is identically distributed in the real execution and the simulated one.



(a) v_1 and v_2 are neighbors of the receiver

(b) v_1 is not a neighbor of the receiver, but v_2 is

Figure 20: The various settings in Case 1, Item 5, where w_i is the center.

Case 2: P_{v^*} is the center. Consider an arbitrary triplet of neighbors, $\{w_1, w_2, w_3\} \subseteq \mathcal{N}(v^*)$. For every $i \in \{1,2,3\}$ and every pair $v_1, v_2 \in [2n+1]$, we will examine the value $\mathbf{M}_{w_i}[v_1, v_2]$ by considering the following cases:

1. If $\{v_1, v_2\} \cap \{v^*, w_i\} \neq \emptyset$ or $v_1 = v_2$, it holds that $\mathbf{M}_{w_i}[v_1, v_2] = 0$. The same holds in the simulation.

2. If $\{v_1, v_2\} \cap \mathcal{N}(w_i) = \emptyset$, then $\mathbf{M}_{w_i}[v_1, v_2]$ is sampled uniformly at random. The same holds in the simulation.

3. If $\{v_1, v_2\} \cap \mathcal{N}(w_i) = \{v_j\}$ with $j \in \{1,2\}$ (i.e., $|\{v_1, v_2\} \cap \mathcal{N}(w_i)| = 1$), then, according to Step 3, $\mathbf{M}_{w_i}[v_j, v_{3-j}]$ is set as:

$$\begin{aligned}\mathbf{M}_{w_i}[v_j, v_{3-j}] &= \sigma^{v_{3-j}}_{v_j, w_i} \oplus \sigma^{v_j}_{v_{3-j}, w_i} \oplus m_{w_i} \\ &= \sigma^{v_{3-j}}_{v_j, w_i} \oplus \sigma^{v_j}_{v_{3-j}, w_i} \oplus 0 \\ &= \mathbf{C}^{\text{3-nbr}}_{v_j}[w_i, v_{3-j}] \oplus 0 \oplus 0 \\ &= \mathbf{C}^{\text{3-nbr}}_{v_j}[w_i, v_{3-j}],\end{aligned}$$

   where the second equality holds since the non-center w_i sets $m_{w_i} = 0$ in Step 3, the third equality holds since $v_j \in \mathcal{N}(w_i)$ so $\sigma^{v_{3-j}}_{v_j, w_i} = \mathbf{C}^{\text{3-nbr}}_{v_j}[w_i, v_{3-j}]$, and since $v_{3-j} \notin \mathcal{N}(w_i)$ so $\sigma^{v_j}_{v_{3-j}, w_i} = 0$. Note that the value $\mathbf{C}^{\text{3-nbr}}_{v_j}[w_i, v_{3-j}]$ is sampled uniformly at random, and is known only to v_j (who sampled it) and w_i who received it, and is used only once in this specific message.

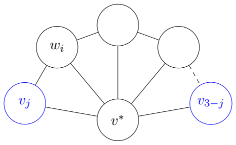


Figure 21: The setting in Case 2, Item 3, where the corrupted receiver P_{v^*} is the center, and v_j is a neighbor of w_i but v_{3-j} is not a neighbor of w_i.

4. If $|\{v_1, v_2\} \cap \mathcal{N}(w_i)| = 2$, then according to Step 3, $\mathbf{M}_{w_i}[v_1, v_2]$ is formed as:

$$\begin{aligned}\mathbf{M}_{w_i}[v_1, v_2] &= \sigma^{v_2}_{v_1, w_i} \oplus \sigma^{v_1}_{v_2, w_i} \oplus m_{w_i} \\ &= \sigma^{v_2}_{v_1, w_i} \oplus \sigma^{v_1}_{v_2, w_i} \oplus 0 \\ &= \mathbf{C}^{\text{3-nbr}}_{v_1}[w_i, v_2] \oplus \mathbf{C}^{\text{3-nbr}}_{v_2}[w_i, v_1],\end{aligned}$$

   where the second equality holds since the non-center w_i sets $m_{w_i} = 0$ in Step 3, and the third equality holds since $\{v_1, v_2\} \subseteq \mathcal{N}(w_i)$ so $\sigma^{v_2}_{v_1, w_i} = \mathbf{C}^{\text{3-nbr}}_{v_1}[w_i, v_2]$, and $\sigma^{v_1}_{v_2, w_i} = \mathbf{C}^{\text{3-nbr}}_{v_2}[w_i, v_1]$.

   Note that the values $\mathbf{C}^{\text{3-nbr}}_{v_1}[w_i, v_2]$ and $\mathbf{C}^{\text{3-nbr}}_{v_2}[w_i, v_1]$ were sent to w_i from the corresponding v_1 or v_2 at Step 2. Party v_1 samples $\mathbf{C}^{\text{3-nbr}}_{v_1}[w_i, v_2]$ uniformly at random and sends it only to w_i (different parties get different $\mathbf{C}^{\text{3-nbr}}_{v_1}[w_i, v_2]$). Similarly, Party v_2 samples $\mathbf{C}^{\text{3-nbr}}_{v_2}[w_i, v_1]$ uniformly at random and sends it only to w_i. Therefore, both $\mathbf{C}^{\text{3-nbr}}_{v_1}[w_i, v_2]$ and $\mathbf{C}^{\text{3-nbr}}_{v_2}[w_i, v_1]$ are not known to P_{v^*}, and are not used in other values.

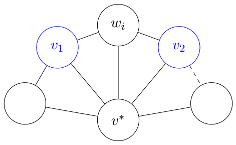


Figure 22: The setting in Case 2, Item 4, where the corrupted receiver P_{v^*} is the center, and both v_1 and v_2 are neighbors of w_i.

This concludes the proof of Theorem 4.3. □

In Theorem 4.3 we have shown a 1-IT-THB protocol for the "fixed-size" wheel graph-class. With minor changes to the protocol (including the blue instruction lines) wheel graphs of different sizes are also captured.

Definition 4.4 (augmented wheel graph). *Let $k, n \in \mathbb{N}$ such that $4 \leq k \leq n$. The* augmented wheel graph *$W_{k,n}$ consists of the wheel graph W_k together with $n + 1 - (k + 1) = n - k$ isolated nodes.*

Definition 4.5 (augmented wheel graph-class). *Let $k, n \in \mathbb{N}$ such that $4 \leq k \leq n$. The* augmented wheel graph-class, *denoted by $\mathcal{G}_{\mathsf{wheel}}(k, n)$, is the isomorphically closed graph class associated with $W_{k,n}$.*

Corollary 4.6 (IT-THB for wheel graph-class of variable size). *Let $n \in \mathbb{N}$ with $n \geq 4$, and let $\mathcal{G} \subseteq \bigcup_{k=4}^{n} \mathcal{G}_{\mathsf{wheel}}(k, n)$. The protocol π_{wheel} (including the blue instruction lines) is a perfectly secure IT-THB protocol against a single semi-honest corruption with respect to $\mathcal{G}$.*

4.2 1-IT-THB for Admissible Subgraphs

We proceed by carefully combining the protocols for friendship graphs and wheel graphs into a single protocol suitable for arbitrary admissible graphs. We start by defining admissible graphs as a special form of star-embedded graphs (which will be used also in Section 5.2).

Definition 4.7 (star-embedded graph). *Let $n \in \mathbb{N}$ such that $n \geq 4$. A graph G is an n-*star-embedded graph *if the following requirements hold:*

- *There exists $k \in \mathbb{N}$ such that $4 \leq k \leq n$ and G is a subgraph of $W_{k,n}$ with $|\mathcal{V}(G)| = n + 1$.*
- *G contains a single, main connected component of (V, E), such that $|V| \geq 5$ and there exists a node $v \in V$ whose degree is $\deg(v) = |V| - 1$.*
- *All other connected components are singletons; that is, for every node $u \in \mathcal{V}(G) \setminus V$ it holds that $\deg(u) = 0$.*

Definition 4.8 (admissible graph). *Let $n \in \mathbb{N}$ such that $n \geq 4$. A graph G is an n-*admissible graph *if G is an n-star-embedded graph and the following requirement holds:*

- *Let (V, E) be the main connected component of G. For every node $u \in V$, if $\deg(u) \neq |V| - 1$, then $\deg(u) \in \{2, 3\}$.*

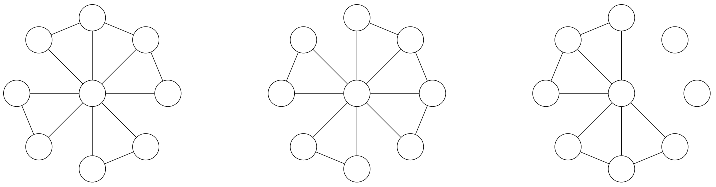


Figure 23: Examples of admissible subgraphs of W_8.

Definition 4.9 (star-embedded graph-class)**.** *Let $n \in \mathbb{N}$. A graph-class $\mathcal{G}(n)$ is an n-*star-embedded graph-class *if $\mathcal{G}(n)$ is not empty, and every $G \in \mathcal{G}(n)$ is an n-star-embedded graph, whose nodes are labeled by unique identities from $[n+1]$, and if $\mathcal{G}(n)$ is isomorphically closed.*

Definition 4.10 (admissible graph-class)**.** *Let $n \in \mathbb{N}$. A graph-class $\mathcal{G}_{\mathsf{admis}}(n)$ is an n-*admissible graph-class *if $\mathcal{G}_{\mathsf{admis}}(n)$ is an n-star-embedded graph-class, and every $G \in \mathcal{G}_{\mathsf{admis}}(n)$ is an n-admissible graph.*

We proceed to present an IT-THB protocol for an admissible graph-class. At a high level, the protocol combines both the friendship protocol and the wheel protocol simultaneously; however, care must be taken to ensure that the receiver "destroys" the irrelevant execution. An overview can be found in the introduction (Section 1.2).

Notations used in the protocol. The notations in this protocol is a combination of the notations in the friendship protocol ($\pi_{\mathsf{friendship}}$, Figure 15) and the wheel protocol (π_{wheel}, Figure 17).

- As in $\pi_{\mathsf{friendship}}$, the *blinding terms* are represented by the vectors $\boldsymbol{b}_u^{\mathsf{mul}}$ and $\boldsymbol{b}_u^{\mathsf{add}}$.
- As in $\pi_{\mathsf{friendship}}$, the *suitable offset* values $\beta_u^{\text{2-nbr}}$ are used to hide the structure of the graph if the receiver is the center. As opposed to $\pi_{\mathsf{friendship}}$, the suitable offset values are also used if the receiver has three neighbors, meaning that the "friendship subexecution" should be meaningless.
- As in $\pi_{\mathsf{friendship}}$, every neighbor u of the receiver sends to the receiver a vector $\boldsymbol{s}_u$ in the last step of the protocol.
- As in π_{wheel}, the matrix of *correlated values* is represented as $\mathbf{C}_u^{\text{3-nbr}}$.
- As in π_{wheel}, every neighbor u of the receiver sends to the receiver a matrix $\mathbf{M}_u$ in the last step of the protocol.

Theorem 4.11 (IT-THB for admissible graph-class)**.** *Let $n \in \mathbb{N}$ and let $\mathcal{G}_{\mathsf{admis}}(n)$ be an n-admissible graph-class. Then, Protocol π_{admis} (defined in Figure 24) is a perfectly secure IT-THB protocol against a single semi-honest corruption with respect to $\mathcal{G}_{\mathsf{admis}}(n)$.*

Proof. First, note that the protocol is well-defined since there exists a center node $v \in V$ with degree $\deg(v) = |V| - 1$, and the center is unique since $|V| \geq 5$. Further, for every non-center node $u \in V$ it holds that $\deg(u) \in \{2, 3\}$ as required by the protocol. We proceed to prove *correctness* and *security.*

Protocol $\pi_{\mathsf{admis}}(n, \mathsf{P}_S)$

Auxiliary input: A binary field $\mathbb{F}_q$ such that $q > n+1$ and an n-admissible graph-class $\mathcal{G}_{\mathsf{admis}}(n)$.

Input: The sender P_S, with $S \in [n+1]$, holds an input $m \in \{0,1\}$.

Hybrid model: The protocol is defined in the $\mathcal{F}_{\mathsf{graph}}^{\mathcal{G}_{\mathsf{admis}}(n)}$-hybrid model.

The protocol:

- Each party P_v sends an initialization message to $\mathcal{F}_{\mathsf{graph}}^{\mathcal{G}_{\mathsf{admis}}(n)}$ and receives its neighbor-set $\mathcal{N}(v)$.
- Repeat for each potential receiver P_R with $R \in [n+1] \setminus \{S\}$:

1. *Forwarding the message to the center node.* The sender P_S sends m to its neighbors.
   If P_S is isolated, the center acts as if received $m = 0$ from P_S.
2. *Generating blinding terms.* Every non-receiver $\mathsf{P}_u \neq \mathsf{P}_R$ uniformly samples from $\mathbb{F}_q^{n+1}$ two blinding vectors $\boldsymbol{b}_u^{\mathsf{mul}}$ and $\boldsymbol{b}_u^{\mathsf{add}}$ of size $n+1$, and sends $(\boldsymbol{b}_u^{\mathsf{mul}}[v], \boldsymbol{b}_u^{\mathsf{add}}[v])$ to every P_v with $v \in \mathcal{N}(u)$.
   In case P_u is the center, it plays in its head every isolated P_v; that is, P_u samples on behalf of P_v two blinding vectors $\boldsymbol{b}_v^{\mathsf{mul}}$ and $\boldsymbol{b}_v^{\mathsf{add}}$ and sends $(\boldsymbol{b}_v^{\mathsf{mul}}[u], \boldsymbol{b}_v^{\mathsf{add}}[u])$ to itself.
3. *Generating correlated values.* Every party $\mathsf{P}_u \neq \mathsf{P}_R$ samples a matrix $\mathbf{C}_u^{\text{3-nbr}} \leftarrow \mathbb{F}_q^{(n+1)\times(n+1)}$. Next, P_u sends to every P_v with $v \in \mathcal{N}(u) \setminus \{R\}$ the row corresponding to v in $\mathbf{C}_u^{\text{3-nbr}}$.
   In case P_u is the center, it plays in its head every isolated P_v; that is, P_u samples on behalf of P_v a matrix $\mathbf{C}_v^{\text{3-nbr}} \leftarrow \mathbb{F}_q^{(n+1)\times(n+1)}$ and sends the row corresponding to u to itself.
4. *Generating suitable offsets.* Party P_R generates values $\beta_u^{\text{2-nbr}}$ for each of its neighbors $u \in \mathcal{N}(R)$, and sends $\beta_u^{\text{2-nbr}}$ to P_u, as follows:
   - If $|\mathcal{N}(R)| = 0$, i.e., R is isolated, it does not send any messages.
   - If $|\mathcal{N}(R)| = 2$, sample $\beta^{\text{2-nbr}} \leftarrow \mathbb{F}_q$ and set $\beta_u^{\text{2-nbr}} := \beta^{\text{2-nbr}}$ for every $u \in \mathcal{N}(R)$.
   - Otherwise, P_R is the center party or $\mathcal{N}(R) = 3$, sample independently at random $\beta_u^{\text{2-nbr}} \leftarrow \mathbb{F}_q$, for each $u \in \mathcal{N}(R)$, conditioned on all values being distinct.
5. *Forwarding m to P_R.* In this step, the communication is by neighbors of P_R who talk to P_R.
   Every party P_u, with $u \in \mathcal{N}(R)$, proceeds as follows:
   - If u is the center node, set $m_u := m$; else, set $m_u := 0$.
   - Initialize a matrix $\mathbf{M}_u$ of size $(n+1)^2$ to zeros. For every distinct $v_1, v_2 \in [n+1] \setminus \{R, u\}$:
     (a) If $\{v_1, v_2\} \cap \mathcal{N}(u) = \emptyset$, sample $x \leftarrow \mathbb{F}_q$ and set both $\mathbf{M}_u[v_1, v_2] := x$ and $\mathbf{M}_u[v_2, v_1] := x$;
     (b) Otherwise, set $\mathbf{M}_u[v_1, v_2] := \sigma_{v_1,u}^{v_2} \oplus \sigma_{v_2,u}^{v_1} \oplus m_u$, where
       * For $i \in \{1,2\}$, if u is the center set $\sigma_{v_i,u}^{v_{3-i}} := \mathbf{C}_u^{\text{3-nbr}}[v_i, v_{3-i}]$; else, if $v_i \in \mathcal{N}(u)$ set $\sigma_{v_i,u}^{v_{3-i}} := \mathbf{C}_{v_i}^{\text{3-nbr}}[u, v_{3-i}]$, and if $v_i \notin \mathcal{N}(u)$ set $\sigma_{v_i,u}^{v_{3-i}} := 0$.
   - Initialize a vector of zeros $\boldsymbol{s}_u$ of size $n+1$. For every $v \in [n+1] \setminus \{R, u\}$, if P_u is not the center and $v \notin \mathcal{N}(u)$, set $\boldsymbol{s}_u[v] \leftarrow \mathbb{F}_q$ to be a random value; otherwise, set

$$\boldsymbol{s}_u[v] := (\beta_u^{\text{2-nbr}} \cdot (\boldsymbol{b}_u^{\mathsf{mul}}[v] \oplus \boldsymbol{b}_v^{\mathsf{mul}}[u])) \oplus (\boldsymbol{b}_u^{\mathsf{add}}[v] \oplus \boldsymbol{b}_v^{\mathsf{add}}[u]) \oplus m_u.$$

   Finally, P_u sends to P_R the matrix $\mathbf{M}_u$ and the vector $\boldsymbol{s}_u$.
6. P_R *output.*
   - If P_R is the center, output m.
   - If $|\mathcal{N}(R)| = 3$, denote $\mathcal{N}(R) = \{v_1, v_2, v_3\}$ and output $\mathbf{M}_{v_1}[v_2, v_3] \oplus \mathbf{M}_{v_2}[v_1, v_3] \oplus \mathbf{M}_{v_3}[v_1, v_2]$.
   - If $|\mathcal{N}(R)| = 2$, denote $\mathcal{N}(R) = \{v, u\}$ and output $\boldsymbol{s}_{v_1}[v_2] \oplus \boldsymbol{s}_{v_2}[v_1]$.
   - If $|\mathcal{N}(R)| = 0$, output 0.

Figure 24: Information-theoretic THB over $\mathcal{G}_{\mathsf{admis}}$. The instruction lines in blue are only needed when considering admissible graphs with variable size (i.e., when some nodes are isolated).

Correctness. We will show that all parties in the same connected component of P_S output its message m. In case P_R is a neighbor of P_S, then in the first round P_R receives m and sets m as its output. In case P_R is *not* a neighbor of P_S, then P_R has either 2 or 3 neighbors. We consider each case separately.

- If $|\mathcal{N}(R)| = 3$, denote $\mathcal{N}(R) = \{v_1, v_2, u\}$; by the structure of admissible graphs, one of these neighbors, say P_u, is the center. The output in this case is defined as

  $$\begin{aligned}
  &\mathbf{M}_{v_1}[v_2, u] \oplus \mathbf{M}_{v_2}[v_1, u] \oplus \mathbf{M}_u[v_1, v_2] = \\
  &= \left(\sigma^u_{v_2,v_1} \oplus \sigma^{v_2}_{u,v_1} \oplus m_{v_1}\right) \oplus \left(\sigma^u_{v_1,v_2} \oplus \sigma^{v_1}_{u,v_2} \oplus m_{v_2}\right) \oplus \left(\sigma^{v_2}_{v_1,u} \oplus \sigma^{v_1}_{v_2,u} \oplus m_u\right) \\
  &= \left((0 \oplus \sigma^{v_2}_{u,v_1}) \oplus m_{v_1}\right) \oplus \left((0 \oplus \sigma^{v_1}_{u,v_2}) \oplus m_{v_2}\right) \oplus \left((\sigma^{v_2}_{v_1,u} \oplus \sigma^{v_1}_{v_2,u}) \oplus m_u\right) \\
  &= \left((0 \oplus \mathbf{C}^{\text{3-nbr}}_u[v_1, v_2]) \oplus m_{v_1}\right) \oplus \left((0 \oplus \mathbf{C}^{\text{3-nbr}}_u[v_2, v_1]) \oplus m_{v_2}\right) \oplus \left((\mathbf{C}^{\text{3-nbr}}_u[v_1, v_2] \oplus \mathbf{C}^{\text{3-nbr}}_u[v_2, v_1]) \oplus m_u\right) \\
  &= (\mathbf{C}^{\text{3-nbr}}_u[v_1, v_2] \oplus \mathbf{C}^{\text{3-nbr}}_u[v_2, v_1] \oplus \mathbf{C}^{\text{3-nbr}}_u[v_1, v_2] \oplus \mathbf{C}^{\text{3-nbr}}_u[v_2, v_1]) \oplus (m_{v_1} \oplus m_{v_2} \oplus m_u) \\
  &= (0 \oplus 0) \oplus (0 \oplus 0 \oplus m) = m.
  \end{aligned}$$

  The first equality holds by the way $\mathbf{M}_{v_1}[v_2, u]$, $\mathbf{M}_{v_2}[v_1, u]$, and $\mathbf{M}_u[v_1, v_2]$ are defined in Step 5. The second equality holds because the center node u is the common neighbor of v_1 and v_2, which are not adjacent parties (by the structure of admissible graphs), so $\sigma^u_{v_1,v_2} = \sigma^u_{v_2,v_1} = 0$. The third equality holds since u is the center so $\sigma^{v_{3-i}}_{v_i,u} = \mathbf{C}^{\text{3-nbr}}_u[v_i, v_{3-i}]$ for $i \in \{1, 2\}$, and since v_1 and v_2 are non-center, hence $\sigma^{v_{3-i}}_{u,v_i} = \mathbf{C}^{\text{3-nbr}}_u[v_i, v_{3-i}]$. The fourth equality holds by commutativity, and the last equality holds because non-center nodes set $m_{v_1} = m_{v_2} = 0$.

- If $|\mathcal{N}(R)| = 2$, denote $\mathcal{N}(R) = \{v, u\}$; again, one of these neighbors, say P_u, is the center. The output in this case is defined to be

  $$\begin{aligned}
  &\boldsymbol{s}_v[u] \oplus \boldsymbol{s}_u[v] = \\
  &= \left((\beta^{\text{2-nbr}}_v \cdot (\boldsymbol{b}^{\mathsf{mul}}_v[u] \oplus \boldsymbol{b}^{\mathsf{mul}}_u[v])) \oplus (\boldsymbol{b}^{\mathsf{add}}_u[v] \oplus \boldsymbol{b}^{\mathsf{add}}_v[u]) \oplus m_v\right) \\
  &\quad \oplus \left((\beta^{\text{2-nbr}}_u \cdot (\boldsymbol{b}^{\mathsf{mul}}_u[v] \oplus \boldsymbol{b}^{\mathsf{mul}}_v[u])) \oplus (\boldsymbol{b}^{\mathsf{add}}_v[u] \oplus \boldsymbol{b}^{\mathsf{add}}_u[v]) \oplus m_u\right) \\
  &= \left(\beta^{\text{2-nbr}} \cdot (\boldsymbol{b}^{\mathsf{mul}}_v[u] \oplus \boldsymbol{b}^{\mathsf{mul}}_u[v] \oplus \boldsymbol{b}^{\mathsf{mul}}_u[v] \oplus \boldsymbol{b}^{\mathsf{mul}}_v[u])\right) \\
  &\quad \oplus (\boldsymbol{b}^{\mathsf{add}}_u[v] \oplus \boldsymbol{b}^{\mathsf{add}}_v[u] \oplus \boldsymbol{b}^{\mathsf{add}}_v[u] \oplus \boldsymbol{b}^{\mathsf{add}}_u[v]) \oplus (m_v \oplus m_u) \\
  &= (\beta^{\text{2-nbr}} \cdot 0) \oplus (0 \oplus 0) \oplus (0 \oplus m) = m,
  \end{aligned}$$

  where the first equality holds by the way $\boldsymbol{s}_v[u]$ and $\boldsymbol{s}_u[v]$ are defined in Step 5; the second equality holds by Step 4 and because the receiver P_R, which is not the center, sets $\beta^{\text{2-nbr}}_v = \beta^{\text{2-nbr}}_u = \beta^{\text{2-nbr}}$; and the last equality holds because the non-center neighbor P_v sets $m_v = 0$ in Step 5.

Security. We proceed to prove security. Let P_{v^*} with $v^* \in [n+1]$ denote the corrupted party. We will construct a simulator Sim that given the neighbor-set of P_{v^*} generates a simulated view for P_{v^*} that is identically distributed as its view in a real execution of the protocol. As we consider semi-honest security, and as broadcast is a deterministic functionality, this implies that the environment's output is identically distributed in the real and ideal computations.

The simulator Sim runs P_{v^*} in its head. Initially, P_{v^*} sends an initialization message to $\mathcal{F}^{\mathcal{G}}_{\mathsf{graph}}$; the simulator forwards this message to $\mathcal{W}^{\mathcal{G}}_{\mathsf{graph\text{-}info}}(\mathcal{F}_{\mathsf{bc}}(\mathsf{P}_S))$ and sends the response $\mathcal{N}(v^*)$

to P_{v^*}. Next, in case the sender is corrupted, i.e., $\mathsf{P}_{v^*} = \mathsf{P}_S$, the simulator sends its input m to $\mathcal{W}^{\mathcal{G}}_{\mathsf{graph\text{-}info}}(\mathcal{F}_{\mathsf{bc}}(\mathsf{P}_S))$; regardless, Sim receives the message m as the output from $\mathcal{W}^{\mathcal{G}}_{\mathsf{graph\text{-}info}}(\mathcal{F}_{\mathsf{bc}}(\mathsf{P}_S))$ (formally, Sim sends the "empty input" for a non-sender corrupted party). Further, the simulator instructs $\mathcal{W}^{\mathcal{G}}_{\mathsf{graph\text{-}info}}(\mathcal{F}_{\mathsf{bc}}(\mathsf{P}_S))$ to set the output of every party that is not in the connected component of P_S to be 0.

Next, Sim simulates an RMT instance towards the corrupted parties, for every potential receiver P_R with $R \in [n+1] \setminus \{S\}$.

- To simulate Step 1, if the sender is corrupted, Sim receives m on behalf of every honest P_v for which $v \in \mathcal{N}(S)$. If the sender is honest and a neighbor of P_{v^*}, then Sim sends m to P_{v^*}. If the sender is isolated and the center is honest, then Sim simulates the center as if receiving $m = 0$.
- To simulate Step 2, if P_{v^*} is not the receiver, Sim samples for every honest neighbor P_u with $u \in \mathcal{N}(v^*)$ random blinding terms $\boldsymbol{b}^{\mathsf{mul}}_u[v^*], \boldsymbol{b}^{\mathsf{add}}_u[v^*] \leftarrow \mathbb{F}_q$ and sends $(\boldsymbol{b}^{\mathsf{mul}}_u[v^*], \boldsymbol{b}^{\mathsf{add}}_u[v^*])$ to P_{v^*} on behalf of P_u. In addition, Sim receives $(\boldsymbol{b}^{\mathsf{mul}}_{v^*}[u], \boldsymbol{b}^{\mathsf{add}}_{v^*}[u])$ from P_{v^*} on behalf of P_u.
- To simulate Step 3, if P_{v^*} is not the receiver, for every honest neighbor P_u with $u \in \mathcal{N}(v^*)$, Sim samples uniformly at random from $\mathbb{F}_q^{(n+1)\times(n+1)}$ correlated values for a square matrix $\mathbf{C}^{\text{3-nbr}}_u$ and sends the corresponding row vector to P_{v^*} on behalf of P_u. In addition, Sim receives the vector $\mathbf{C}^{\text{3-nbr}}_{v^*}[u]$ from P_{v^*} on behalf of P_u.
- To simulate Step 4, if P_{v^*} is a neighbor of P_R, i.e., $R \in \mathcal{N}(v^*)$, the simulator samples at random $\beta^{\text{2-nbr}}_{v^*} \leftarrow \mathbb{F}_q$ and sends $\beta^{\text{2-nbr}}_{v^*}$ to P_{v^*}. Else, If $v^* = R$, the simulator receives $\beta^{\text{2-nbr}}_u$ from P_{v^*} on behalf of P_u for every $u \in \mathcal{N}(v^*)$.
- To simulate Step 5, if P_{v^*} is a neighbor of the receiver, i.e., $v^* \in \mathcal{N}(R)$, then Sim receives from P_{v^*} the vector $\boldsymbol{s}_{v^*}$ and a matrix $\mathbf{M}_{v^*}$ on behalf of P_R.

  If P_{v^*} is the receiver, we consider three cases:

  1. If P_{v^*} is the center node, then for each $u \neq v^*$ the simulator initializes a vector $\boldsymbol{s}_u$ of size $n+1$ and a matrix $\mathbf{M}_u$ of size $(n+1) \times (n+1)$ to zeros. For $v \in [n+1] \setminus \{R, u\}$ the simulator sets $\boldsymbol{s}_u[v] \leftarrow \mathbb{F}_q$. For $v_1, v_2 \in [n+1] \setminus \{u, v^*\}$ such that $v_1 \neq v_2$, the simulator samples $x \leftarrow \mathbb{F}_q$ at random and sets $\mathbf{M}_u[v_1, v_2] = \mathbf{M}_u[v_2, v_1] = x$. Next, Sim sends the vector $\boldsymbol{s}_u$ and the matrix $\mathbf{M}_u$ to P_{v^*} on behalf of P_u.
  2. If P_{v^*} has two neighbors, denote $\mathcal{N}(v^*) = \{w_1, w_2\}$. The simulator initializes two vectors $\boldsymbol{s}_{w_1}$ and $\boldsymbol{s}_{w_2}$ each of size $n+1$ and two matrices $\mathbf{M}_{w_1}$ and $\mathbf{M}_{w_2}$ of size $(n+1)^2$ with zeros. It samples from $\mathbb{F}_q$ and sets the values $\boldsymbol{s}_{w_1}[w_2]$, $\boldsymbol{s}_{w_2}[w_1]$ conditioned on $\boldsymbol{s}_{w_1}[w_2] \oplus \boldsymbol{s}_{w_2}[w_1] = m$. For every $v \in [n+1] \setminus \{v^*, w_1, w_2\}$ the simulator samples $\boldsymbol{s}_{w_1}[v], \boldsymbol{s}_{w_2}[v] \leftarrow \mathbb{F}_q$ at random. In addition, for every $v_1, v_2 \in [n+1] \setminus \{v^*, w_1, w_2\}$ such that $v_1 \neq v_2$ the simulator samples $x_1, x_2 \leftarrow \mathbb{F}_q$ at random and sets

     $$\begin{aligned} \mathbf{M}_{w_1}[v_1, v_2] =& \mathbf{M}_{w_1}[v_2, v_1] = x_1 \\ \mathbf{M}_{w_2}[v_1, v_2] =& \mathbf{M}_{w_2}[v_2, v_1] = x_2. \end{aligned}$$

     Eventually, Sim sends the adversary the vectors $\boldsymbol{s}_{w_i}$ and the matrices $\mathbf{M}_{w_i}$ for $i \in \{1, 2\}$.

3. If P_{v^*} has three neighbors, denote $\mathcal{N}(v^*) = \{w_1, w_2, w_3\}$. The simulator initializes the matrices $\mathbf{M}_{w_1}$, $\mathbf{M}_{w_2}$ and $\mathbf{M}_{w_3}$ each of size $(n+1) \times (n+1)$ to zeros, then the simulator randomly samples the values $x_1, x_2, x_3 \leftarrow \mathbb{F}_q$ conditioned on $x_1 \oplus x_2 \oplus x_3 = m$ and sets

$$\begin{aligned}\mathbf{M}_{w_1}[w_2, w_3] =& \mathbf{M}_{w_1}[w_3, w_2] = x_1 \\ \mathbf{M}_{w_2}[w_1, w_3] =& \mathbf{M}_{w_2}[w_3, w_1] = x_2 \\ \mathbf{M}_{w_3}[w_2, w_1] =& \mathbf{M}_{w_3}[w_1, w_2] = x_3.\end{aligned}$$

Let $i \in \{1,2,3\}$, for every $v_1, v_2 \in [n+1] \setminus \{v^*, w_i\}$ such that $|\{v_1, v_2\} \cap \mathcal{N}(v^*)| < 2$ and $v_1 \neq v_2$ the simulator samples at random $x'_i \leftarrow \mathbb{F}_q$, and sets

$$\mathbf{M}_{w_i}[v_1, v_2] = \mathbf{M}_{w_i}[v_2, v_1] = x'_i.$$

In addition, for each w_i the simulator initializes a vector $\boldsymbol{s}_{w_i}$ of size $n+1$ to zeros, and for every $v \in [n+1] \setminus \{v^*, w_i\}$ the simulator samples $\boldsymbol{s}_{w_i}[v] \leftarrow \mathbb{F}_q$ at random. Eventually, the simulator sends the adversary the vector $\boldsymbol{s}_{w_i}$ matrix $\mathbf{M}_{w_i}$.

We proceed to show that the view of P_{v^*} in the simulated protocol is identically distributed as its view in a real execution of the protocol. Note that the simulation mirrors the protocol behavior except for the last step. Therefore, we need to analyze only the last step (i.e., Step 5), where v^* is the receiver. we do so by considering three cases: in Lemma 4.12 we consider the case where P_{v^*} is the center; in Lemma 4.15 the case where P_{v^*} has two neighbors; and in Lemma 4.18 the case where P_{v^*} has three neighbors.

Lemma 4.12. *Suppose that P_{v^*} is the center. Then, the view of Adv in Step 5 of the real execution of $\pi_{\mathsf{admis}}(n, \mathsf{P}_S)$ and the view of Adv in Step 5 of the simulated execution (as part of the ideal computation) are identically distributed.*

Proof. Note that the view of Adv (which is essentially the view of P_{v^*}) can be decomposed into to $\{\boldsymbol{s}_w\}_{w \in \mathcal{N}(v^*)}$ (which are related to the "friendship subexecution") and $\{\mathbf{M}_w\}_{w \in \mathcal{N}(v^*)}$ (which are related to the "wheel subexecution"). In Claims 4.13 and 4.14 we will show that each part is identically distributed on its own; next, we will show that these messages are distributed independently, and the proof will follow.

Claim 4.13. *The partial view of P_{v^*} consisting of the vectors $\{\boldsymbol{s}_w\}_{w \in \mathcal{N}(v^*)}$ is identically distributed in the real execution and in the simulated execution.*

Proof. Consider an arbitrary pair of neighbors $\{w_1, w_2\} \subseteq \mathcal{N}(v^*)$. For every $i \in \{1,2\}$ we will analyze the vector $\boldsymbol{s}_{w_i}$ by examining the value $\boldsymbol{s}_{w_i}[v]$ for every $v \in [n+1]$. Consider the following cases:

1. If $v \in \{v^*, w_i\}$, then $\boldsymbol{s}_{w_i}[v] = 0$ both in the real execution and in the simulated execution.

2. If $v \notin \mathcal{N}(w_i)$, then $\boldsymbol{s}_{w_i}[v]$ is sampled uniformly at random both in the real execution and in the simulated execution.

3. If $v \in \mathcal{N}(w_i) \setminus \{v^*\}$, then by Step 5 of the real protocol, the value $\boldsymbol{s}_{w_i}[v]$ is formed as

$$\begin{aligned}\boldsymbol{s}_{w_i}[v] &= (\beta^{\text{2-nbr}}_{w_i} \cdot (\boldsymbol{b}^{\mathsf{mul}}_{w_i}[v] \oplus \boldsymbol{b}^{\mathsf{mul}}_{v}[w_i])) \oplus (\boldsymbol{b}^{\mathsf{add}}_{w_i}[v] \oplus \boldsymbol{b}^{\mathsf{add}}_{v}[w_i]) \oplus m_{w_i} \\ &= (\beta^{\text{2-nbr}}_{w_i} \cdot (\boldsymbol{b}^{\mathsf{mul}}_{w_i}[v] \oplus \boldsymbol{b}^{\mathsf{mul}}_{v}[w_i])) \oplus (\boldsymbol{b}^{\mathsf{add}}_{w_i}[v] \oplus \boldsymbol{b}^{\mathsf{add}}_{v}[w_i]),\end{aligned}$$

where the second equality holds since P_{v^*} is the center, hence w_i sets m_{w_i} to zero.

The values $\boldsymbol{b}_{w_i}^{\mathsf{mul}}[v]$, $\boldsymbol{b}_{v}^{\mathsf{mul}}[w_i]$, $\boldsymbol{b}_{w_i}^{\mathsf{add}}[v]$ and $\boldsymbol{b}_{v}^{\mathsf{add}}[w_i]$ are sent between w_i and v, and those are the only two parties that know and use those values. Party P_v uses those values only when forming $\boldsymbol{s}_v[w_i]$ as

$$\begin{aligned}\boldsymbol{s}_v[w_i] &= (\beta_v^{\text{2-nbr}} \cdot (\boldsymbol{b}_{v}^{\mathsf{mul}}[w_i] \oplus \boldsymbol{b}_{w_i}^{\mathsf{mul}}[v])) \oplus (\boldsymbol{b}_{v}^{\mathsf{add}}[w_i] \oplus \boldsymbol{b}_{w_i}^{\mathsf{add}}[v]) \oplus m_v \\ &= (\beta_v^{\text{2-nbr}} \cdot (\boldsymbol{b}_{v}^{\mathsf{mul}}[w_i] \oplus \boldsymbol{b}_{w_i}^{\mathsf{mul}}[v])) \oplus (\boldsymbol{b}_{v}^{\mathsf{add}}[w_i] \oplus \boldsymbol{b}_{w_i}^{\mathsf{add}}[v]),\end{aligned}$$

where, again, the second equality holds since P_{v^*} is the center, so v sets m_v to zero. Denote,

$$b^{\mathsf{mul}} = \boldsymbol{b}_{w}^{\mathsf{mul}}[v] \oplus \boldsymbol{b}_{v}^{\mathsf{mul}}[w] \quad \text{and} \quad b^{\mathsf{add}} = \boldsymbol{b}_{v}^{\mathsf{add}}[w] \oplus \boldsymbol{b}_{w}^{\mathsf{add}}[v].$$

Note that b^{mul} and b^{add} are uniformly distributed, since $\boldsymbol{b}_{w_i}^{\mathsf{mul}}[v]$, $\boldsymbol{b}_{v}^{\mathsf{mul}}[w_i]$, $\boldsymbol{b}_{w_i}^{\mathsf{add}}[v]$ and $\boldsymbol{b}_{v}^{\mathsf{add}}[w_i]$ are sampled uniformly at random at Step 2. In addition, note that the values $\beta_{w_i}^{\text{2-nbr}}$ and $\beta_v^{\text{2-nbr}}$ are *known* to P_{v^*} (as P_{v^*} sampled and sent them), and by Step 4 we are guaranteed that $\beta_{w_i}^{\text{2-nbr}} \neq \beta_v^{\text{2-nbr}}$ because P_{v^*} is the center.

Looking back at $\boldsymbol{s}_{w_i}[v]$ and $\boldsymbol{s}_v[w_i]$, it holds that

$$\boldsymbol{s}_{w_i}[v] = (\beta_{w_i}^{\text{2-nbr}} \cdot b^{\mathsf{mul}}) \oplus b^{\mathsf{add}} \quad \text{and} \quad \boldsymbol{s}_v[w_i] = (\beta_v^{\text{2-nbr}} \cdot b^{\mathsf{mul}}) \oplus b^{\mathsf{add}}.$$

From the eyes of P_{v^*}, these are two linear equations with two unknowns b^{mul} and b^{add}, which are uniformly distributed. These equations are solvable since $\beta_{w_i}^{\text{2-nbr}} \neq \beta_v^{\text{2-nbr}}$. Therefore, from the view of P_{v^*}, the values $\boldsymbol{s}_{w_i}[v]$ and $\boldsymbol{s}_v[w_i]$ are distributed as a pair of values sampled uniformly at random from $\mathbb{F}_q$.

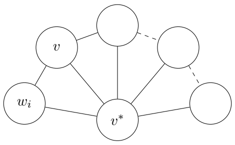


Figure 25: Illustration of the scenario described in Item 3 of Claim 4.13.

This concludes the proof of Claim 4.13. □

Claim 4.14. *The partial view of* P_{v^*} *consisting of the matrices* $\{\mathbf{M}_w\}_{w\in\mathcal{N}(v^*)}$ *is identically distributed in the real execution and in the simulated execution.*

Proof. Consider an arbitrary neighbor $w \in \mathcal{N}_G(v^*)$. For every $v_1, v_2 \in [n+1]$, we will examine the value $\mathbf{M}_w[v_1, v_2]$ by considering the following cases:

1. If $\{v_1, v_2\} \cap \{v^*, w\} \neq \emptyset$ or $v_1 = v_2$ the value $\mathbf{M}_w[v_1, v_2] = 0$. The same holds in the simulation.
2. If $\{v_1, v_2\} \cap \mathcal{N}(w) = \emptyset$, then $\mathbf{M}_w[v_1, v_2]$ is sampled uniformly at random, both in the simulation and in the real execution.

3. If $|\{v_1, v_2\} \cap \mathcal{N}(w)| = 1$. Denote $j \in \{1, 2\}$ such that $\{v_1, v_2\} \cap \mathcal{N}(w) = \{v_j\}$. Using Step 5 to disassemble $\mathbf{M}_w[v_{3-j}, v_j]$ value;

$$\begin{aligned}\mathbf{M}_w[v_j, v_{3-j}] &= \sigma^{v_{3-j}}_{v_j,w} \oplus \sigma^{v_j}_{v_{3-j},w} \oplus m_w \\ &= \sigma^{v_{3-j}}_{v_j,w} \oplus \sigma^{v_j}_{v_{3-j},w} \oplus 0 \\ &= \mathbf{C}^{\text{3-nbr}}_{v_j}[w, v_{3-j}] \oplus 0 \oplus 0 \\ &= \mathbf{C}^{\text{3-nbr}}_{v_j}[w, v_{3-j}].\end{aligned}$$

The second equality holds since v^* is the center, therefore, the non-center w sets $m_w = 0$ in Step 5. The third equality holds since $v_j \in \mathcal{N}(w)$ so $\sigma^{v_{3-j}}_{v_j,w} = \mathbf{C}^{\text{3-nbr}}_{v_j}[w, v_{3-j}]$, and since $v_{3-j} \notin \mathcal{N}(w)$ so $\sigma^{v_j}_{v_{3-j},w} = 0$.
Note that the value $\mathbf{C}^{\text{3-nbr}}_{v_j}[w, v_{3-j}]$ is sampled uniformly at random, and is known only to v_j (who sampled it) and w who received it, and is used only once in this specific message.

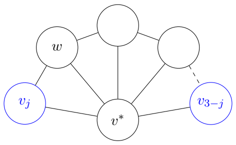


Figure 26: Illustration of the scenario described in Item 3 of Claim 4.14. Here, v_j is a neighbor of w and v_{3-j} is not.

4. If $|\{v_1, v_2\} \cap \mathcal{N}(w)| = 2$, using Step 5 to disassemble $\mathbf{M}_w[v_1, v_2]$ value;

$$\begin{aligned}\mathbf{M}_w[v_1, v_2] &= \sigma^{v_2}_{v_1,w} \oplus \sigma^{v_1}_{v_2,w} \oplus m_w \\ &= \sigma^{v_2}_{v_1,w} \oplus \sigma^{v_1}_{v_2,w} \oplus 0 \\ &= \mathbf{C}^{\text{3-nbr}}_{v_1}[w, v_2] \oplus \mathbf{C}^{\text{3-nbr}}_{v_2}[w, v_1].\end{aligned}$$

The second equality holds since v^* is the center, therefore, the non-center w sets $m_w = 0$ in Step 5. The third equality holds since $\{v_1, v_2\} \subseteq \mathcal{N}(w)$ so $\sigma^{v_2}_{v_1,w} = \mathbf{C}^{\text{3-nbr}}_{v_1}[w, v_2]$, and $\sigma^{v_1}_{v_2,w} = \mathbf{C}^{\text{3-nbr}}_{v_2}[w, v_1]$.
Note that the values $\mathbf{C}^{\text{3-nbr}}_{v_1}[w, v_2]$, $\mathbf{C}^{\text{3-nbr}}_{v_2}[w, v_1]$ were sent to w from the corresponding v_1 or v_2 at Step 3. Party v_1 samples $\mathbf{C}^{\text{3-nbr}}_{v_1}[w, v_2]$ uniformly at random and sends it only to w (different parties get different $\mathbf{C}^{\text{3-nbr}}_{v_1}[w, v_2]$). Similarly, party v_2 samples $\mathbf{C}^{\text{3-nbr}}_{v_2}[w, v_1]$ uniformly at random and sends it only to w. Both $\mathbf{C}^{\text{3-nbr}}_{v_1}[w, v_2]$ and $\mathbf{C}^{\text{3-nbr}}_{v_2}[w, v_1]$ are not known to P_{v^*}, and are not used in other values.

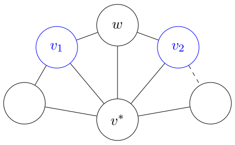


Figure 27: Illustration of the scenario described in Item 4 of Claim 4.14. Here, both v_1 and v_2 are neighbors of w.

This concludes the proof of Claim 4.14. □

Finally, since P_{v^*} is the center, in the real execution, the messages $\{\boldsymbol{s}_w\}_{w \in \mathcal{N}(v^*)}$ and $\{\mathbf{M}_w\}_{w \in \mathcal{N}(v^*)}$ are not dependent on the sender's message m, and are therefore distributed as a product distribution of uniformly random values from $\mathbb{F}_q$ (or the constant value 0 for the coordinates corresponding to sending neighbor w and the receiver v^*). The same holds in the simulated execution; hence, the views are identically distributed. This concludes the proof of Lemma 4.12. □

Lemma 4.15. *Suppose that P_{v^*} has two neighbors. Then, the view of Adv in Step 5 of the real execution of $\pi_{\mathsf{admis}}(n, \mathsf{P}_S)$ and the view of Adv in Step 5 of the simulated execution (as part of the ideal computation) are identically distributed.*

Proof. Note that the view of Adv (which is essentially the view of P_{v^*}) can be decomposed into to $\{\boldsymbol{s}_w\}_{w \in \mathcal{N}(v^*)}$ (which are related to the "friendship subexecution") and $\{\mathbf{M}_w\}_{w \in \mathcal{N}(v^*)}$ (which are related to the "wheel subexecution"). In Claims 4.16 and 4.17 we will show that each part is identically distributed on its own; next, we will show that these messages are distributed independently, and the proof will follow.

Claim 4.16. *The partial view of P_{v^*} consisting of the matrices $\{\mathbf{M}_w\}_{w \in \mathcal{N}(v^*)}$ is identically distributed in the real execution and in the simulated execution.*

Proof. Denote $\mathcal{N}(v^*) = \{w_1, w_2\}$. Let $i \in \{1, 2\}$. For every $v_1, v_2 \in [n+1]$, we will examine how the value $\mathbf{M}_{w_i}[v_1, v_2]$ is formed in the real execution, to show that it is identically distributed as in the simulation.

1. If $\{v_1, v_2\} \cap \{v^*, w_i\} \neq \emptyset$ or $v_1 = v_2$ the value $\mathbf{M}_{w_i}[v_1, v_2] = 0$. The same holds in the simulation.

2. If $\{v_1, v_2\} \cap \mathcal{N}(w_i) = \emptyset$, then $\mathbf{M}_{w_i}[v_1, v_2]$ is sampled uniformly at random. The same holds in the simulation.

3. If w_i is not the center and $\{v_1, v_2\} \cap \mathcal{N}(w_i) = \{v_j\}$ with $j \in \{1, 2\}$ (i.e., $|\{v_1, v_2\} \cap \mathcal{N}(w_i)| = 1$, see Figure 28), then according to Step 5, $\mathbf{M}_{w_i}[v_{3-j}, v_j]$ is set as:

$$\begin{aligned} \mathbf{M}_{w_i}[v_{3-j}, v_j] &= \sigma^{v_j}_{v_{3-j}, w_i} \oplus \sigma^{v_{3-j}}_{v_j, w_i} \oplus m_{w_i} \\ &= \sigma^{v_j}_{v_{3-j}, w_i} \oplus \sigma^{v_{3-j}}_{v_j, w_i} \oplus 0 \\ &= 0 \oplus \mathbf{C}^{\text{3-nbr}}_{v_j}[w_i, v_{3-j}], \end{aligned}$$

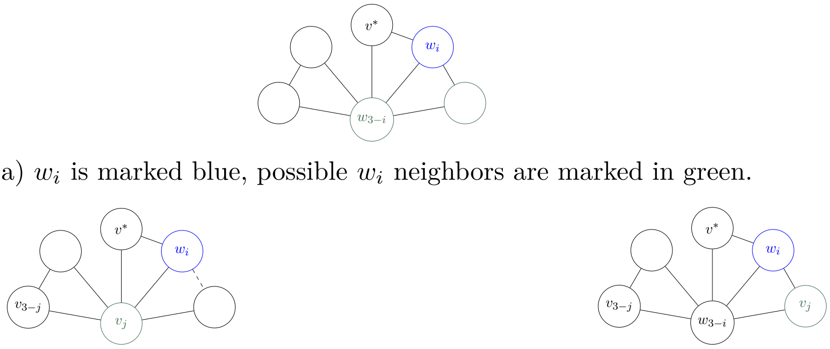
where the second equality holds since a non-center w_i sets $m_{w_i} = 0$, and the third equality holds since a non-center w_i sets $\sigma^{v_{3-j}}_{v_j, w_i} = \mathbf{C}^{\text{3-nbr}}_{v_j}[w_i, v_{3-j}]$ for $v_j \in \mathcal{N}(w_i)$, and sets $\sigma^{v_j}_{v_{3-j}, w_i} = 0$ for $v_{3-j} \notin \mathcal{N}(w_i)$.

The only other party that can use the value $\mathbf{C}^{\text{3-nbr}}_{v_j}[w_i, v_{3-j}]$ is the center;[8] if v_j is the center, the relevant value $\mathbf{M}_{v_j}[w_i, v_{3-j}]$ is formed as:

$$\begin{aligned} \mathbf{M}_{v_j}[w_i, v_{3-j}] &= \sigma^{v_{3-j}}_{w_i, v_j} \oplus \sigma^{w_i}_{v_{3-j}, v_j} \oplus m_{v_j} \\ &= \sigma^{v_{3-j}}_{w_i, v_j} \oplus \sigma^{w_i}_{v_{3-j}, v_j} \oplus m \\ &= \mathbf{C}^{\text{3-nbr}}_{v_j}[w_i, v_{3-j}] \oplus \mathbf{C}^{\text{3-nbr}}_{v_j}[v_{3-j}, w_i] \oplus m. \end{aligned}$$

[8] We highlight some values with blue and green colors to match the illustration in Figure 28.

In $\mathbf{M}_{v_j}[w_i, v_{3-j}]$ the value $\mathbf{C}^{\text{3-nbr}}_{v_j}[v_{3-j}, w_i]$ is a value the center sends to party v_{3-j}. Since P_{v^*} has only 2 neighbors, party v_{3-j} cannot be a neighbor of P_{v^*}, hence this value $\mathbf{C}^{\text{3-nbr}}_{v_j}[v_{3-j}, w_i]$ is not used in any other message that is part of the receiver's view and is identically distributed as the value in the simulation which is sampled uniformly at random.



(a) w_i is marked blue, possible w_i neighbors are marked in green.

(b) First option; where v_j is the center

(c) Second option; where v_j is not the center

Figure 28: In case receiver is non-center, illustration of the values a non-center w_i party will send P_{v^*}, as shown in Item 3 of Claim 4.16.

4. If w_i is not the center and $\{v_1, v_2\} \subseteq \mathcal{N}(w_i)$, then according to Step 5, $\mathbf{M}_{w_i}[v_{3-j}, v_j]$ is set as:

$$\begin{aligned}\mathbf{M}_{w_i}[v_1, v_2] &= \sigma^{v_2}_{v_1,w_i} \oplus \sigma^{v_1}_{v_2,w_i} \oplus m_{w_i}\\ &= \sigma^{v_2}_{v_1,w_i} \oplus \sigma^{v_1}_{v_2,w_i} \oplus 0\\ &= \mathbf{C}^{\text{3-nbr}}_{v_1}[w_i, v_2] \oplus \mathbf{C}^{\text{3-nbr}}_{v_2}[w_i, v_1] \oplus 0,\end{aligned}$$

where the second equality holds since a non-center w_i sets $m_{w_i} = 0$, and the third equality holds according to Step 5: $\sigma^{v_2}_{v_1,w_i} = \mathbf{C}^{\text{3-nbr}}_{v_1}[w_i, v_2]$ and $\sigma^{v_1}_{v_2,w_i} = \mathbf{C}^{\text{3-nbr}}_{v_2}[w_i, v_1]$.

From the structure of admissible graphs, since w_i is not the center, either v_1 or v_2 is not a neighbor of the receiver; assume without loss of generality that v_1 is not a neighbor of R. Consider the value $\mathbf{C}^{\text{3-nbr}}_{v_1}[w_i, v_2]$: the only time when this value is communicated is when party v_1 sends it to party w_i at Step 3. During the simulation, Sim samples $\mathbf{M}_{w_i}[v_1, v_2]$ uniformly at random and independently of all other values; hence, $\mathbf{M}_{w_i}[v_1, v_2]$ is identically distributed in the real execution and the simulated one.

This concludes the proof of Claim 4.16. □

Claim 4.17. *The partial view of P_{v^*} consisting of the vectors $\{\boldsymbol{s}_w\}_{w \in \mathcal{N}(v^*)}$ is identically distributed in the real execution and in the simulated execution.*

Proof. Denote $\mathcal{N}(v^*) = \{w_1, w_2\}$. Let $i \in \{1, 2\}$, and $v \in [n+1]$, we show that the values $\boldsymbol{s}_{w_i}[v]$ in the real execution are identically distributed as in the simulation.

1. If $v \in \{v^*, w_i\}$ the value $\boldsymbol{s}_{w_i}[v] = 0$. The same holds in the simulation.

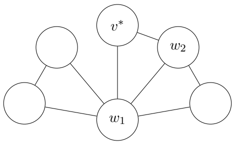
2. If $v \notin \mathcal{N}(w_i)$ then in both the real execution and the simulation the value $\boldsymbol{s}_{w_i}[v]$ is sampled uniformly at random. Hence, the view identically distributed.

3. For $v = w_{3-i}$, the value $\boldsymbol{s}_{w_i}[w_{3-i}]$ in the real execution satisfies $\boldsymbol{s}_{w_1}[w_2] \oplus \boldsymbol{s}_{w_2}[w_1] = m$. Indeed, the simulation guarantees all values are sampled conditioned on $\boldsymbol{s}_{w_1}[w_2] \oplus \boldsymbol{s}_{w_2}[w_1] = m$.

4. For $v \in \mathcal{N}(w_i) \setminus \{w_{3-i}\}$, in the real execution the value is formed by:

$$\boldsymbol{s}_{w_i}[v] = (\beta^{\text{2-nbr}}_{w_i} \cdot (\boldsymbol{b}^{\mathsf{mul}}_{w_i}[v] \oplus \boldsymbol{b}^{\mathsf{mul}}_{v}[w_i])) \oplus (\boldsymbol{b}^{\mathsf{add}}_{w_i}[v] \oplus \boldsymbol{b}^{\mathsf{add}}_{v}[w_i]) \oplus m_{w_i},$$

   where $\boldsymbol{b}^{\mathsf{add}}_{w_i}[v]$ and $\boldsymbol{b}^{\mathsf{add}}_{v}[w_i]$ are sampled uniformly at random, and are only known to P_{w_i} and P_v. Note that $v \notin \mathcal{N}[v^*]$, therefore $v \notin \{w_1, w_2\}$. Particularly, P_{v^*} is not aware of those blinding terms, and each term is used only once, therefore the value $\boldsymbol{s}_{w_i}[v]$ is uniformly distributed. In the simulation, the value is sampled uniformly at random by the simulation, hence the view is identically distributed.



This concludes the proof of Claim 4.17. □

Finally, the view of P_{v^*} can be decomposed into $\{\boldsymbol{s}_w\}_{w \in \mathcal{N}(v^*)}$ and $\{\mathbf{M}_w\}_{w \in \mathcal{N}(v^*)}$. Since P_{v^*} has two neighbors, we have shown in Claim 4.16 that all the values in $\{\mathbf{M}_w\}_{w \in \mathcal{N}(v^*)}$ are distributed as a product of uniformly random values from $\mathbb{F}_q$ (or the constant value 0 for the coordinates corresponding to sending neighbor w and the receiver v^*). By Claim 4.17, the values in $\{\boldsymbol{s}_w\}_{w \in \mathcal{N}(v^*)}$ are also distributed as a product of uniformly random values from $\mathbb{F}_q$ (or the constant value 0 for the coordinates corresponding to sending neighbor w and the receiver v^*), except for the values that construct the message eventually. The same holds in the simulated execution; hence, the views are identically distributed. This concludes the proof of Lemma 4.15. □

Lemma 4.18. *Suppose that P_{v^*} has three neighbors. Then, the view of Adv in Step 5 of the real execution of $\pi_{\mathsf{admis}}(n, \mathsf{P}_S)$ and the view of Adv in Step 5 of the simulated execution (as part of the ideal computation) are identically distributed.*

Proof. Note that the view of Adv (which is essentially the view of P_{v^*}) can be decomposed into to $\{\boldsymbol{s}_w\}_{w \in \mathcal{N}(v^*)}$ (which are related to the "friendship subexecution") and $\{\mathbf{M}_w\}_{w \in \mathcal{N}(v^*)}$ (which are related to the "wheel subexecution"). Like before, in Claims 4.19 and 4.20 we will show that each part is identically distributed on its own; next, we will show that these messages are distributed independently, and the proof will follow.

Claim 4.19. *The partial view of P_{v^*} consisting of the matrices $\{\mathbf{M}_w\}_{w \in \mathcal{N}(v^*)}$ is identically distributed in the real execution and in the simulated execution.*

Proof. Denote $\mathcal{N}(v^*) = \{w_1, w_2, w_3\}$. Let $i \in \{1,2,3\}$, and $v_1, v_2 \in [n+1]$, we will examine how the values $\mathbf{M}_{w_i}[v_1, v_2]$ are formed in the real execution, to show that they are identically distributed as in the simulation.

1. If $\{v_1, v_2\} \cap \{R, w_i\} \neq \emptyset$ or $v_1 = v_2$ the value $\mathbf{M}_{w_i}[v_1, v_2] = 0$. The same holds in the simulation.

2. If $\{v_1, v_2\} \cap \mathcal{N}(w_i) = \emptyset$, then $\mathbf{M}_{w_i}[v_1, v_2]$ is sampled uniformly at random. The same holds in the simulation.

3. If w_i is not the center and $|\{v_1, v_2\} \cap \mathcal{N}(w_i)| = 1$, let $j \in \{1, 2\}$ and $\{v_1, v_2\} \cap \mathcal{N}(w_i) = \{v_j\}$ then according to Step 1, $\mathbf{M}_{w_i}[v_{3-j}, v_j]$ is set as:

$$\begin{aligned}\mathbf{M}_{w_i}[v_{3-j}, v_j] &= \sigma^{v_j}_{v_{3-j},w_i} \oplus \sigma^{v_{3-j}}_{v_j,w_i} \oplus m_{w_i}\\ &= \sigma^{v_j}_{v_{3-j},w_i} \oplus \sigma^{v_{3-j}}_{v_j,w_i} \oplus 0\\ &= 0 \oplus \mathbf{C}^{\text{3-nbr}}_{v_j}[w_i, v_{3-j}],\end{aligned}$$

   where the second equality holds since a non-center w_i sets $m_{w_i} = 0$, and the third equality holds since a non-center w_i sets $\sigma^{v_{3-j}}_{v_j,w_i} = \mathbf{C}^{\text{3-nbr}}_{v_j}[w_i, v_{3-j}]$ for $v_j \in \mathcal{N}(w_i)$, and sets $\sigma^{v_j}_{v_{3-j},w_i} = 0$ for $v_{3-j} \notin \mathcal{N}(w_i)$.

   The only other party that can use the value $\mathbf{C}^{\text{3-nbr}}_{v_j}[w_i, v_{3-j}]$ is the center;[9] if v_j is the center, the relevant value $\mathbf{M}_{v_j}[w_i, v_{3-j}]$ is formed as:

$$\begin{aligned}\mathbf{M}_{v_j}[w_i, v_{3-j}] &= \sigma^{v_{3-j}}_{w_i,v_j} \oplus \sigma^{w_i}_{v_{3-j},v_j} \oplus m_{v_j}\\ &= \sigma^{v_{3-j}}_{w_i,v_j} \oplus \sigma^{w_i}_{v_{3-j},v_j} \oplus m\\ &= \mathbf{C}^{\text{3-nbr}}_{v_j}[w_i, v_{3-j}] \oplus \mathbf{C}^{\text{3-nbr}}_{v_j}[v_{3-j}, w_i] \oplus m.\end{aligned}$$

   In $\mathbf{M}_{v_j}[w_i, v_{3-j}]$ the value $\mathbf{C}^{\text{3-nbr}}_{v_j}[v_{3-j}, w_i]$ is a value the center sends to party v_{3-j}, if v_{3-j} is a neighbor of the receiver, then $\{w_i, v_j, v_{3-j}\} \subseteq \{w_1, w_2, w_3\}$ else, v_{3-j} is not a neighbor of the receiver, hence this value $\mathbf{C}^{\text{3-nbr}}_{v_j}[v_{3-j}, w_i]$ is not used in any other message that is part of the receiver's view and is identically distributed as the value in the simulation which is sampled uniformly at random.

4. If w_i is not the center and $\{v_1, v_2\} \subseteq \mathcal{N}(w_i)$ then according to Step 1, $\mathbf{M}_{w_i}[v_{3-j}, v_j]$ is formed by:

$$\begin{aligned}\mathbf{M}_{w_i}[v_1, v_2] &= \sigma^{v_2}_{v_1,w_i} \oplus \sigma^{v_1}_{v_2,w_i} \oplus m_{w_i}\\ &= \sigma^{v_2}_{v_1,w_i} \oplus \sigma^{v_1}_{v_2,w_i} \oplus 0\\ &= \mathbf{C}^{\text{3-nbr}}_{v_1}[w_i, v_2] \oplus \mathbf{C}^{\text{3-nbr}}_{v_2}[w_i, v_1] \oplus 0,\end{aligned}$$

   the second equality holds since w_i is not the center, $m_{w_i} = 0$, the third equality holds according to Step 5; $\sigma^{v_2}_{v_1,w_i} = \mathbf{C}^{\text{3-nbr}}_{v_1}[w_i, v_2]$ and $\sigma^{v_1}_{v_2,w_i} = \mathbf{C}^{\text{3-nbr}}_{v_2}[w_i, v_1]$.

   By the structure of admissible graphs, since w_i is not the center, either v_1 or v_2 is not a neighbor of the receiver; assume without loss of generality that v_1 is not a neighbor of R. Consider the value $\mathbf{C}^{\text{3-nbr}}_{v_1}[w_i, v_2]$: the only time when this value is communicated is when party v_1 sends it to party w_i at Step 3. During the simulation, Sim samples $\mathbf{M}_{w_i}[v_1, v_2]$ uniformly at random and independently of all other values; hence, $\mathbf{M}_{w_i}[v_1, v_2]$ is identically distributed in the real execution and the simulated one.

[9] This setting is illustrated in Figure 18, and we highlight some values with blue and green to match the illustration.

5. If w_i is the center, then in particular $\{v_1, v_2\} \subseteq \mathcal{N}(w_i)$, and according to Step 1, $\mathbf{M}_{w_i}[v_1, v_2]$ is formed as:

$$\begin{aligned}\mathbf{M}_{w_i}[v_1, v_2] &= \sigma^{v_2}_{v_1,w_i} \oplus \sigma^{v_1}_{v_2,w_i} \oplus m_{w_i} \\ &= \sigma^{v_2}_{v_1,w_i} \oplus \sigma^{v_1}_{v_2,w_i} \oplus m \\ &= \mathbf{C}^{\text{3-nbr}}_{w_i}[v_1, v_2] \oplus \mathbf{C}^{\text{3-nbr}}_{w_i}[v_2, v_1] \oplus m,\end{aligned}$$

where the second equality holds since w_i is the center, so $m_{w_i} = m$, and the third equality holds since w_i is the center so $\sigma^{v_2}_{v_1,w_i} = \mathbf{C}^{\text{3-nbr}}_{w_i}[v_1, v_2]$ and $\sigma^{v_1}_{v_2,w_i} = \mathbf{C}^{\text{3-nbr}}_{w_i}[v_2, v_1]$.

Note that the case where $\{v_1, v_2, w_i\} \subseteq \{w_1, w_2, w_3\}$ is handled above, at the beginning of Case 1. Otherwise, without loss of generality, let $v_1 \notin \{w_1, w_2, w_3\}$; in this case, the value $\mathbf{C}^{\text{3-nbr}}_{w_i}[v_1, v_2]$ is sent from w_i to v_1 in Step 3, and is only used in $\mathbf{M}_{w_i}[v_1, v_2]$ (since v_1 not a neighbor of the receiver). During the simulation, Sim samples $\mathbf{M}_{w_i}[v_1, v_2]$ uniformly at random and independently of all other values; hence, $\mathbf{M}_{w_i}[v_1, v_2]$ is identically distributed in the real execution and the simulated one.

This concludes the proof of Claim 4.19. □

Claim 4.20. *The partial view of P_{v^*} consisting of the vectors $\{\boldsymbol{s}_w\}_{w\in\mathcal{N}(v^*)}$ is identically distributed in the real execution and in the simulated execution.*

Proof. Note that the view of P_{v^*} can be derived from the view of the center. Therefore, the claim follows from Claim 4.13. □

Finally, the view of P_{v^*} can be decomposed into $\{\boldsymbol{s}_w\}_{w\in\mathcal{N}(v^*)}$ and $\{\mathbf{M}_w\}_{w\in\mathcal{N}(v^*)}$. Since P_{v^*} has three neighbors, we have shown that in Claim 4.20 that all the values in $\{\boldsymbol{s}_w\}_{w\in\mathcal{N}(v^*)}$ are distributed as a product of uniformly random values from $\mathbb{F}_q$ (or the constant value 0 for the coordinates corresponding to sending neighbor w and the receiver v^*). For the values in $\{\mathbf{M}_w\}_{w\in\mathcal{N}(v^*)}$, by Claim 4.19 they are also distributed as a product of uniformly random values from $\mathbb{F}_q$ (or the constant value 0 for the coordinates corresponding to sending neighbor w and the receiver v^*), except for the values that construct the message eventually. The same holds in the simulated execution; hence, the views are identically distributed. This concludes the proof of Lemma 4.18. □

This concludes the proof of Theorem 4.11. □

5 Lower Bounds and Characterization

In Section 5.1, we present our lower bounds, and in Section 5.2, we characterize which subgraphs of the wheel containing an embedded start support 1-IT-THB and which require key agreement.

5.1 Lower Bounds

We proceed to prove two simple impossibility results for 1-IT-THB. In Section 5.1.1, we show that a topology-hiding broadcast protocol for a class containing an admissible graph-class together with the star graph-class of the same size, necessitates key agreement. In Section 5.1.2 we show an analogue result for graphs containing a tail node and a non-tail node. We prove the bounds by a direct reduction to graph classes that were shown to imply key agreement in [BBC+20] via the phantom-jump technique (see Section 1.2.2). These bounds are given with respect to the weaker definition of topology-hiding broadcast, IND-CTA security (see Section 2.1.2); this establishes stronger results.

5.1.1 1-THB on Admissible Subgraphs and Stars Requires Key Agreement

The first bound is accomplished by a reduction to the class $\mathcal{G}_{\mathsf{triangle}}$ (see Figure 8). We refer the reader to Section 1.2.2 for an overview of the ideas. As we consider graphs with variable size, i.e., with isolated nodes, we first define the *augmented star graph* in the spirit of augmented friendship graphs (Definition 3.4).

Definition 5.1 (star graph)**.** *Let $n \in \mathbb{N}$ such that $n \geq 2$. The* star graph *S_n consists of $n+1$ nodes: a center node of degree n, and n nodes of degree 1 connected to the center.*

Definition 5.2 (augmented star graph)**.** *Let $k, n \in \mathbb{N}$ such that $2 \leq k \leq n$. The* augmented star graph *$S_{k,n}$ consists of the star graph S_k together with $n+1-(k+1) = n-k$ isolated nodes.*

Definition 5.3 (augmented star graph-class)**.** *Let $k, n \in \mathbb{N}$ such that $2 \leq k \leq n$. The* augmented star graph-class*, denoted by $\mathcal{G}_{\mathsf{star}}(k,n)$, is the isomorphically closed graph class associated with $S_{k,n}$.*

Lemma 5.4. *Let $n \in \mathbb{N}$ such that $n \geq 5$ and let $\mathcal{G}_1$ and $\mathcal{G}_2$ be two n-star-embedded graph-classes (as per Definition 4.9). Assume that the following holds:*

- *The class $\mathcal{G}_1$ is an n-admissible graph-class (as per Definition 4.10); in particular, there exists a graph $G_1 \in \mathcal{G}_1$, with a main connected component (V_1, E_1), such that for every node $u \in V_1$, if $\deg(u) \neq |V_1| - 1$, then $\deg(u) \in \{2, 3\}$.*
- *The class $\mathcal{G}_2$ contains an augmented star graph-class; in particular, there exists a graph $G_2 \in \mathcal{G}_2$, with a main connected component (V_2, E_2), such that for every $u \in V_2$, if $\deg(u) \neq |V_2| - 1$, then $\deg(u) = 1$.*
- *The set of non-isolated nodes in G_1 and G_2 (i.e., in their main connected components) is the same, i.e., $V_1 = V_2$.*

Then, if there exists a 1-IND-CTA-secure broadcast protocol for the class $\mathcal{G}_1 \cup \mathcal{G}_2$, there exists a key-agreement protocol.

Proof. Consider the graphs $G_1 \in \mathcal{G}_1$ and $G_2 \in \mathcal{G}_2$ from the theorem statement. Denote the center node of the main connected component in G_1 (and G_2) by u. Since G_1 is admissible, there exist two non-center nodes v_1 and v_2 that are connected by an edge. Since $\mathcal{G}_1$ is isomorphically closed, assume (without loss of generality) that the nodes v_1, v_2, and u are labeled in G_1 with ① , ② , and ③ , respectively. Next, since $\mathcal{G}_2$ is isomorphically closed, assume (without loss of generality) that the nodes v_1, v_2, and u in G_2 are also labeled with ① , ② , and ③ , respectively. Finally, since $\mathcal{G}_2$ is isomorphically closed, there exists a graph $G_3 \in \mathcal{G}_2$ with the same main connected component (V_2, E_2) as G_2, in which the nodes v_1, v_2, and u are labeled with ③ , ② , and ① , respectively.

Consider a partition of the $n+1$ nodes into three sets: the first set consists of the node v_1, the second set consists of the node u, and the third set consists of all other nodes (and in particular includes v_2). Note that for each of the graphs G_1, G_2, and G_3, the nodes labeled by ①, ②, and ③ appear in different sets; denote by $\mathcal{P}_i$ the set that contains ⓘ, for $i \in \{1, 2, 3\}$. Then, the following holds:

- When considering the labeling of G_1, it holds that $\mathcal{P}_1 = \{\textcircled{1}\}$, $\mathcal{P}_2 = \{\textcircled{2}, \textcircled{4}, \ldots, \boxed{n+1}\}$, and $\mathcal{P}_3 = \{\textcircled{3}\}$, and there are edges between $\mathcal{P}_1$ and $\mathcal{P}_2$, between $\mathcal{P}_2$ and $\mathcal{P}_3$, and between $\mathcal{P}_1$ and $\mathcal{P}_3$. That is, this is the structure of three sets on a triangle $\mathcal{P}_1$—$\mathcal{P}_2$—$\mathcal{P}_3$—$\mathcal{P}_1$ and $|\mathcal{P}_1| = |\mathcal{P}_3| = 1$.

- When considering the labeling of G_2, it holds that $\mathcal{P}_1 = \{\textcircled{1}\}$, $\mathcal{P}_2 = \{\textcircled{2}, \textcircled{4}, \ldots, \boxed{n+1}\}$, and $\mathcal{P}_3 = \{\textcircled{3}\}$, and there are edges between $\mathcal{P}_3$ and $\mathcal{P}_2$ and between $\mathcal{P}_3$ and $\mathcal{P}_1$, but there is no edge between $\mathcal{P}_1$ and $\mathcal{P}_2$. That is, this is the structure of three sets on a line $\mathcal{P}_2$—$\mathcal{P}_3$—$\mathcal{P}_1$ and $|\mathcal{P}_1| = |\mathcal{P}_3| = 1$.

- When considering the labeling of G_3, it holds that $\mathcal{P}_1 = \{\textcircled{1}\}$, $\mathcal{P}_2 = \{\textcircled{2}, \textcircled{4}, \ldots, \boxed{n+1}\}$, and $\mathcal{P}_3 = \{\textcircled{3}\}$, and there are edges between $\mathcal{P}_1$ and $\mathcal{P}_2$ and between $\mathcal{P}_1$ and $\mathcal{P}_3$, but there is no edge between $\mathcal{P}_2$ and $\mathcal{P}_3$. That is, this is the structure of three sets on a line $\mathcal{P}_2$—$\mathcal{P}_1$—$\mathcal{P}_3$ and $|\mathcal{P}_1| = |\mathcal{P}_3| = 1$.

Therefore, if there exists a 1-IND-CTA-secure broadcast protocol for $\mathcal{G}$, in which node $\textcircled{2}$ is the sender, then there exists an IND-CTA-secure broadcast protocol for $\mathcal{G}_{\mathsf{triangle}}$ against corruptions of $\mathcal{P}_1$ and $\mathcal{P}_3$, in which $\mathcal{P}_2$ is the sender. By [BBC+20, Thm. 3.5], this implies the existence of a key-agreement protocol. □

5.1.2 1-THB on a Graph with a Tail and a Non-tail Requires Key Agreement

The second bound is accomplished by a reduction to the class $\mathcal{G}_{\mathsf{paw}}$ (see Figure 29). This is a four-node graph consisting of a center of degree 3, a tail of degree 1, and two non-tails of degree 2. By [BBC+20, Lem. 7.6], 1-IND-CTA broadcast on this class requires key agreement.

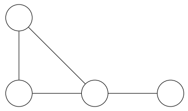


Figure 29: “The Paw” graph

Lemma 5.5. *Let $n \in \mathbb{N}$ such that $n \geq 4$ and let $\mathcal{G}$ be a n-star-embedded graph-class (as per Definition 4.9). Assume that there exists a graph $G \in \mathcal{G}$, with a main connected component (V, E); further, there exist non-center nodes $u, w \in V$ such that $\deg(u) = 1$ and $\deg(w) = 2$.*

Then, if there exists a 1-IND-CTA-secure broadcast protocol for the class $\mathcal{G}$, there exists a key-agreement protocol.

Proof. Consider the following partition of G into four sets:

- The first set $\mathcal{P}_1$ consists of the degree-1 node u.

- The second set $\mathcal{P}_2$ consists of the center node of degree $|V| - 1$.

- The third set $\mathcal{P}_3$ consists of the degree-2 node w.

- The last set, $\mathcal{P}_4$ consists of all other nodes. Note that $\mathcal{P}_4 \neq \emptyset$ as w has a neighbor other than the center.

Then, it holds that the set $\mathcal{P}_1$ is connected only to $\mathcal{P}_2$; the set $\mathcal{P}_2$ is connected to $\mathcal{P}_1$, $\mathcal{P}_3$ and $\mathcal{P}_4$; the set $\mathcal{P}_3$ is connected to $\mathcal{P}_2$, and $\mathcal{P}_4$; and the set $\mathcal{P}_4$ is connected $\mathcal{P}_2$ and $\mathcal{P}_3$. Stated differently, the sets $\mathcal{P}_1$, $\mathcal{P}_2$, $\mathcal{P}_3$, and $\mathcal{P}_4$ form the paw graph, and further, the sets $\mathcal{P}_1$ and $\mathcal{P}_2$ are singletons.

Therefore, if there exists a 1-IND-CTA-secure broadcast protocol for $\mathcal{G}$, then there exists an IND-CTA-secure broadcast protocol for $\mathcal{G}_{\mathsf{paw}}$ against corruptions of $\mathcal{P}_1$ and $\mathcal{P}_2$, in which $\mathcal{P}_3$ is the sender. By [BBC+20, Lem. 7.6], this implies the existence of a key-agreement protocol. □

5.2 Characterization of Wheel Subgraphs With an Embedded Star

We proceed to describe our characterization.

Theorem 5.6 (characterization of IT-THB)**.** *Let $n \in \mathbb{N}$ such that $n \geq 4$ and let $\mathcal{G}$ be an n-star-embedded graph-class (as per Definition 4.9). Then,*

- *if the maximal degree of non-center nodes is 1, i.e., $\mathcal{G} \subseteq \cup_{k \leq n} \mathcal{G}_{\mathsf{star}}(k, n)$ consists only of stars (possibly of different size), or*
- *if the minimal degree of non-center nodes (which are not isolated) is 2 or 3, i.e., $\mathcal{G}$ consists only of n-admissible graphs, or*
- *if the class $\mathcal{G}$ can be partitioned into disjoint classes $\mathcal{G}_1$ and $\mathcal{G}_2$ such that $\mathcal{G}_1$ is an n-admissible graph class, $G_2 \subseteq \cup_{k \leq n} \mathcal{G}_{\mathsf{star}}(k, n)$ consists only of stars, and there do not exist $G_1 \in \mathcal{G}_1$ and $G_2 \in \mathcal{G}_1$ such that the main connected components of G_1 and G_2 are of the same size,*

there exists perfectly secure IT-THB against a single semi-honest corruption over $\mathcal{G}$. Otherwise, THB over $\mathcal{G}$ secure against a single semi-honest corruption exists if and only if key agreement exists.

Proof. Let $\mathcal{G}$ be an n-star-embedded graph-class. We will first show that each case in Theorem 5.6 results with 1-IT-THB, and later that any other graph class requires key agreement. By [BBC+20, Thm. 7.3], key agreement is sufficient for 1-THB over $\mathcal{G}$. We consider the following four cases:

Case 1: The maximal degree of non-center nodes is 1. In this case, $\mathcal{G}$ consists only of augmented stars, i.e., $\mathcal{G} \subseteq \cup_{k \leq n} \mathcal{G}_{\mathsf{star}}(k, n)$. By the structure of star-embedded graphs, every non-isolated party can identify that this is a star topology; further, every non-center node does not know the actual size of the graph. It is immediate to see that the following simple protocol at Figure 30 is 1-IT-THB for $\mathcal{G}$.

Case 2: The minimal degree of non-center nodes is 2 or 3. In this case, $\mathcal{G}$ consists only of admissible graphs, and feasibility follows from Theorem 4.11.

Case 3: $\mathcal{G}$ consists both of stars and admissible graphs, of different sizes. Again, by the structure of star-embedded graphs, each non-isolated party can identify whether the topology of the actual graph is a star or admissible: non-center nodes by checking if their degree is 1, and the center by inspecting its degree (in this case it is guaranteed that for each degree the topology is either admissible or a star). Therefore, all non-isolated parties can non-interactively agree on the THB protocol they should run.

Case 4: Otherwise. In this case, there are two possibilities:

- Either $\mathcal{G}$ contains a two graphs G_1 and G_2 whose main connected components are of the same size, and such that G_1 is admissible and the main connected component of G_2 is a star, in which case the proof follows from Lemma 5.4, or
- There exists a graph $G \in \mathcal{G}$ containing a non-center node of degree 1 and another non-center node of degree 2, in which case the proof follows from Lemma 5.5. □

Protocol $\pi_{\mathcal{G}}(n, \mathsf{P}_S)$

Input: The sender P_S, with $S \in [n+1]$, holds an input $m \in \{0,1\}$.

Hybrid model: The protocol is defined in the $\mathcal{F}^{\mathcal{G}}_{\mathsf{graph}}$-hybrid model.

The protocol:

1. Each party P_v receives its neighbor-set $\mathcal{N}(v)$ from $\mathcal{F}^{\mathcal{G}}_{\mathsf{graph}}$.
2. The sender P_S sends m to its neighbors; if P_S is isolated, the center simulates receiving 0.
3. The center sends m to its neighbors.
4. Each party outputs the message it received; isolated parties output 0.

Figure 30: Information-theoretic 1-THB for a collection of stars

Bibliography

[ALM17] Adi Akavia, Rio LaVigne, and Tal Moran. Topology-hiding computation on all graphs. In *37th Annual International Cryptology Conference (CRYPTO), part I*, pages 447–467, 2017.

[AM17] Adi Akavia and Tal Moran. Topology-hiding computation beyond logarithmic diameter. In *36th Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT), part III*, pages 609–637, 2017.

[BBC$^+$19] Marshall Ball, Elette Boyle, Ran Cohen, Tal Malkin, and Tal Moran. Is information-theoretic topology-hiding computation possible? In *Proceedings of the 17th Theory of Cryptography Conference (TCC), part I*, pages 502–530, 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 *Proceedings of the 18th Theory of Cryptography Conference (TCC), part II*, pages 473–501, 2020.

[BBKM23] Marshall Ball, Alexander Bienstock, Lisa Kohl, and Pierre Meyer. Towards topology-hiding computation from oblivious transfer. In *Proceedings of the 21st Theory of Cryptography Conference (TCC), part I*, pages 349–379, 2023.

[BBMM18] Marshall Ball, Elette Boyle, Tal Malkin, and Tal Moran. Exploring the boundaries of topology-hiding computation. In *37th Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT), part III*, pages 294–325, 2018.

[Can01] Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *Proceedings of the 42nd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 136–145, 2001.

[ERS66] Paul Erdös, Alfréd Rényi, and Vera T. Sós. On a problem of graph theory. *Studia Sci. Math. Hungar.*, 1:215–235, 1966.

[HMTZ16] Martin Hirt, Ueli Maurer, Daniel Tschudi, and Vassilis Zikas. Network-hiding communication and applications to multi-party protocols. In *36th Annual International Cryptology Conference (CRYPTO), part II*, pages 335–365, 2016.

[KMTZ13] Jonathan Katz, Ueli Maurer, Björn Tackmann, and Vassilis Zikas. Universally composable synchronous computation. In *Proceedings of the 10th Theory of Cryptography Conference (TCC)*, pages 477–498, 2013.

[Li22] Shuaishuai Li. Towards practical topology-hiding computation. In Shweta Agrawal and Dongdai Lin, editors, *28th International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT), part I*, pages 588–617, 2022.

[LZM$^+$18] Rio LaVigne, Chen-Da Liu Zhang, Ueli Maurer, Tal Moran, Marta Mularczyk, and Daniel Tschudi. Topology-hiding computation beyond semi-honest adversaries. In *Proceedings of the 16th Theory of Cryptography Conference (TCC), part II*, pages 3–35, 2018.

[LZM$^+$20] Rio LaVigne, Chen-Da Liu Zhang, Ueli Maurer, Tal Moran, Marta Mularczyk, and Daniel Tschudi. Topology-hiding computation for networks with unknown delays. In *Proceedings of the 23rd International Conference on the Theory and Practice of Public-Key Cryptography (PKC), part II*, pages 215–245, 2020.

[MOR15] Tal Moran, Ilan Orlov, and Silas Richelson. Topology-hiding computation. In *Proceedings of the 12th Theory of Cryptography Conference (TCC), part I*, pages 159–181, 2015.

A UC Framework

We present a highly informal overview of the UC framework and refer the reader to [Can01] for further details. The framework is based on the real/ideal paradigm for arguing about the security of a protocol.

The real model. An execution of a protocol π in the real model consists of n PPT *interactive Turing machines* (ITMs) $\mathsf{P}_1, \ldots, \mathsf{P}_n$ representing the parties, along with two additional ITMs: an *adversary* Adv, describing the behavior of the corrupted parties and an *environment* Env, representing the external network environment in which the protocol operates. The environment gives inputs to the honest parties, receives their outputs, and can communicate with the adversary at any point during the execution. It is known that security against the *dummy* adversary (that forwards every message it sees to the environment and acts according to the environment's instructions) is sufficient to achieve security against arbitrary adversaries. Throughout, we consider *synchronous* protocols that proceeds in rounds (this can be formally modeled using the $\mathcal{F}_{\mathsf{sync}}$ functionality [Can01], or using the synchronous framework of [KMTZ13]) and *semi-honest (passive) security* (where corrupted parties continue following the protocol, but reveal their internal state to the adversary). We will consider both *static* corruptions (where Adv chooses the corrupted parties at the onset of the protocol) and *adaptive* corruptions (where Adv can dynamically corrupt parties based on information gathered during the computation), and will explicitly mention at any section which type of corruptions are considered. An t-adversary can corrupt up to t parties during the protocol.

The ideal model. A computation in the ideal model consists of n *dummy* parties $\tilde{\mathsf{P}}_1, \ldots, \tilde{\mathsf{P}}_n$, an *ideal-model adversary* (simulator) Sim, an *environment* Env, and an *ideal functionality* $\mathcal{F}$. As in the real model, the environment gives inputs to the honest (dummy) parties, receives their outputs, and can communicate with the ideal-model adversary at any point during the execution. The dummy parties act as channels between the environment and the ideal functionality, meaning that they send the inputs received from Env to $\mathcal{F}$ and vice-versa. The ideal functionality $\mathcal{F}$ defines the desired behaviour of the computation. $\mathcal{F}$ receives the inputs from the dummy parties, executes the desired computation and sends the output to the parties. The ideal-model adversary does not see the communication between the parties and the ideal functionality, however, Sim can corrupt dummy parties (statically or dynamically) and may communicate with $\mathcal{F}$ according to its specification.

Security definition. We present the definition for static and semi-honest adversaries.

We say that a protocol π UC-realizes (with computational security) an ideal functionality $\mathcal{F}$ in the presence of static semi-honest t-adversaries, if for any PPT static semi-honest t-adversary Adv and any PPT environment Env, there exists a PPT ideal-model t-adversary Sim such that the output distribution of Env in the ideal-model computation of $\mathcal{F}$ with Sim is *computationally indistinguishable* from its output distribution in the real-model execution of π with Adv.

We say that a protocol π UC-realizes (with information-theoretic security) an ideal functionality $\mathcal{F}$ if the above holds even for computationally unbounded Adv, Env, and Sim. In that case the requirement is for the output distribution of Env in the ideal-model computation to be *statistically close* to its output distribution in the real-model execution. If the environment's outputs are identically distributed, we say that π UC-realizes $\mathcal{F}$ with *perfect* security.

The hybrid model. The *$\mathcal{F}$-hybrid model* is a combination of the real and ideal models, it extends the real model with an ideal functionality $\mathcal{F}$. The parties communicate with each other in exactly the same way as in the real model; however, they can also interact with $\mathcal{F}$ as in the ideal model. An important property of the UC framework is that the ideal functionality $\mathcal{F}$ in an $\mathcal{F}$-hybrid model can be replaced with a protocol that UC-realizes $\mathcal{F}$. The composition theorem of [Can01] states the following.

Theorem A.1 ([Can01], informal)**.** *Let ρ be a protocol that UC-realizes $\mathcal{F}$ in the presence of adaptive semi-honest t-adversaries, and let π be a protocol that UC-realizes $\mathcal{G}$ in the $\mathcal{F}$-hybrid model in the presence of adaptive semi-honest t-adversaries. Then, for any* PPT *adaptive semi-honest t-adversary Adv and any* PPT *environment Env, there exists a* PPT *adaptive semi-honest t-adversary Sim in the $\mathcal{F}$-hybrid model such that the output distribution of Env when interacting with the protocol π and Sim is computationally indistinguishable from its output distribution when interacting with the protocol π^ρ (where every call to $\mathcal{F}$ is replaced by an execution of ρ) and Adv in the real model.*