All graph state verification protocols are composably secure

Léo Colisson, Damian Markham, Raja Yehia

All graph state verification protocols are composably secure

Léo Colisson [1], Damian Markham[2], and Raja Yehia [3]

[1]QuSoft and Centrum Wiskunde & Informatica, Science Park 123, 1098 XG Amsterdam, Netherlands
[2]Sorbonne Université, CNRS, LIP6, 4 place Jussieu, F-75005 Paris, France
[3]ICFO-Institut de Ciencies Fotoniques, The Barcelona Institute of Science and Technology, Castelldefels, Spain

February 5, 2024

Abstract

Graph state verification protocols allow multiple parties to share a graph state while checking that the state is honestly prepared, even in the presence of malicious parties. Since graph states are the starting point of numerous quantum protocols, it is crucial to ensure that graph state verification protocols can safely be composed with other protocols, this property being known as *composable security*. Previous works [YDK21] conjectured that such a property could not be proven within the abstract cryptography framework: we disprove this conjecture by showing that *all* graph state verification protocols can be turned into a composably secure protocol with respect to the natural functionality for graph state preparation. Moreover, we show that any *unchanged* graph state verification protocols can also be considered as composably secure for a slightly different, yet useful, functionality. Finally, we show that these two results are optimal, in the sense that any such generic result, considering arbitrary black-box protocols, must either modify the protocol or consider a different functionality.

Along the way, we show a protocol to generalize entanglement swapping to arbitrary graph states that might be of independent interest.

Contents

1 Introduction

Quantum networks enhance today's networks capabilities by providing a higher level of security, based on the inviolable laws of physics, but also by enabling the emergence of new protocols impossible to obtain classically. The spectrum of quantum protocols is wide, starting from quantum teleportation [BBC+93] to delegated computation [BFK09], verifiable computation [FK17, MF18, GKK19], multi-party computation [DNS12, KP17, DGJ+20, GLS+21, CMS23], quantum money [Wie83, BOT+18], anonymous transmission [CW05, UMY+18], copy-protection [Aar09], leader elections and coin flipping [Gan09, BCK+20], e-voting [HZB+06, CDK22], and more.

A large fraction of these protocols, like anonymous transmission protocols [CW05, UMY+18], expect parties to share before the beginning of the protocol a number of fundamental quantum states like Bell pairs, GHZ states, or, more generally, arbitrary *graph states*. This task is typically achieved using a *graph state verification protocol*, whose role is to securely distribute a graph state among all parties.

These graph state verification protocols should typically be resilient to deviations from possibly malicious parties, whether they are controlling the source or not. Such security properties are usually proven in a weak, so called game-based model. In this model, we can only prove guarantees on the final quantum state, but we cannot really obtain any guarantee on the behavior of the protocol when it is composed into other protocols (which is the whole point of graph state verification protocols!), or when the adversary is allowed to run attacks in parallel.

As a consequence, it is often unclear if the security of the original protocol is preserved when the graph state is obtained via a *graph state verification protocol* instead of being honestly generated by a trusted third party, leading to the natural question:

Is it safe to compose any arbitrary protocol with any arbitrary graph state verification protocol?
Is it still secure if the adversary can run multiple attacks in parallel?

The study of the composition of protocols is typically done in a security framework where the notion of *functionality* or *resource* is introduced in order to abstract the properties of a given protocol [Can01, Unr10, MR11a, Mau12]. A functionality can be seen as a trusted party: this way a protocol is said to realize a given functionality if it is impossible to say if we are running the actual protocol or the functionality. With this concept in mind, creating new protocols from sub-protocols is a breeze: we just need to prove that the protocol is secure when the sub-protocol is implemented by a functionality, and we are automatically guaranteed that the protocol will still be secure if the functionality is replaced with any sub-protocol realizing this functionality, even if the adversary is allowed to run attacks in parallel. Composing functionalities is therefore fundamental when designing protocols, since many more advanced protocols are often obtained by composing simpler sub-protocols. This allows to build on previous works and to use functionalities as black-boxes with definite input and outputs.

When using the terminology of these frameworks, the above questions can be reformulated as follows:

Do composable graph state verification protocols exist?

Our results. In this work we answer positively to this interrogation, proving that *any* secure graph state verification protocol is composable. This answers an open question raised in [YDK21] that was suggesting that there might not even exist a single composable state verification protocol.

More specifically:

- We present a method to turn any arbitrary graph state verification protocol, secure in the game-based model, into a composably secure protocol realizing the natural functionality $\mathcal{V}_{|G\rangle}$ for graph state verification. This "compilation" only adds one round of classical communication at the end of the protocol, and mostly preserves the guarantee of the original protocol. More precisely, if the final state obtained in the real protocol is supposed to be ε-close to the target graph state for some notion of closeness, then the protocol ε-realizes $\mathcal{V}_{|G\rangle}$. Our results are expressed in the abstract cryptography framework [MR11b].
- We also show that any *unchanged* graph state verification protocols can also be considered as composably secure for a slightly different, yet useful, functionality.
- We show that it is *impossible* to prove that any arbitrary unchanged protocols realizes $\mathcal{V}_{|G\rangle}$ having only black-box access to the protocol, without either changing the protocol, or the functionality, showing that the above results are optimal. Note that our impossibility result assumes that the simulators has a certain natural structure, which seems hard to avoid when we consider the protocols as black-boxes.
- Along the way, we show a protocol to generalize entanglement swapping to arbitrary graph states, which might be of independent interest. Since graph-state manipulation can be challenging using the usual density matrix formalism, we use scalable ZX-calculus [CK17, CHP19] to prove our result, asserting the relevance of this language for complex graph state operations.

This paper is organized as follows: in Section 2 we start by defining formally graph states verification protocols as well as introducing scalable ZX-calculus and composable security in the Abstract cryptography framework. In Section 3, we then define our ideal functionality outputting graph states and we show the equivalence between generic graph state verification protocols and this ideal functionality. Our ideal functionality applies corrections on the outputted graph state, which might seem unreasonable in a concrete implementation. Hence, in Section 4, we show how to modify graph state verification protocols to realize the ideal functionality without these corrections. Finally, in Section 5, we show how our result applies to two existing graph-state verification protocols.

2 Preliminaries

2.1 Notations

We assume basic familiarity with quantum computing [NC10]. For any subset of index $M \subseteq [n]$, and matrix $G \in \mathbb{Z}_2^{n\times m}$ (resp. $x \in \mathbb{Z}_2^n$), we denote x_M as the vector obtained from x after removing lines not in M. For any quantum gate $\mathbf{X}$, $\mathbf{X}^{x_M}$ will denote the application of $\mathbf{X}$ on all qubits $i \in M$ such that $x_i = 1$. The fidelity $F(\rho, \sigma)$ of two quantum states ρ and σ expressed in term of density matrices is defined as $F(\rho, \sigma) \coloneqq \mathrm{Tr}\sqrt{\rho^{1/2}\sigma\rho^{1/2}}$.

2.2 Graph states

A quantum state $|G\rangle$ is called a graph state if it can be represented by a graph $G = (V, E)$ where the vertices V are $|+\rangle$ states and the edges $E = \{(v_i, v_j)\}_{i=1}^{|E|}$ correspond to controlled $\mathbf{Z}$ gates ($\wedge\mathbf{Z}$) between the vertices v_i and v_j. Thus a graph state can be expressed as

$$|G\rangle = \prod_{(v_i,v_j)\in E} \wedge\mathbf{Z}^{\{v_i,v_j\}} |+\rangle^{\otimes V} \tag{1}$$

As a consequence to this construction, graph states can also be uniquely defined through a set of so-called *stabilizers*. They are operators who leave the graph state unchanged, or in other words, a graph state is the eigenstate of eigenvalue 1 of its stabilizers. For a given graph state $|G\rangle$ with vertices $V = \{v_i\}_{i=1}^{|V|}$, the stabilizers are

$$S_{v_i} = \sigma_X^{v_i} \prod_{v_j \in N(v_i)} \sigma_Z^{v_j} \tag{2}$$

where σ_X and σ_Z are the Pauli matrices and $N(v_i)$ is the neighborhood of v_i. For each vertex v_i, we have that $S_{v_i} |G\rangle = |G\rangle$. The set $\{S_{v_i}\}_{v_i \in V}$ characterizes the graph state $|G\rangle$ and is an equivalent definition to the one of Eq. (1).

Graph states are multipartite entangled states. This means that measuring one of the qubit of a graph state will have an effect on the adjacent qubits of the state. This property is used for example in measurement-based quantum computing [BBD+09] where a computation is done by sequentially measuring the qubits of a universal graph state (e.g. the brickwork state) which propagates through the state. In a communication network setting where a n qubit graph state is shared among n parties, it can be used to create a shared list of correlated bits by asking each party to measure their qubit. This list can then be used in many contexts, for example to create a common secret key among the parties in so-called conference key agreement protocols, the multipartite counterpart of quantum key distribution.

2.3 Scalable ZX-calculus

The ZX calculus [CD08] is a language allowing us to prove equality between quantum operations diagrammatically, using a simple set of rewriting rules between graphs. We present in Appendix A the basic properties of the ZX calculus, as well as one extension called scalable ZX [CHP19], that we use in our proofs. Note however that all equations proven in this article can be checked manually without ZX calculus, and that readers not interested in checking the proof should be able to read this article without any ZX knowledge. We refer the interested reader to [van20] and [CK17] for more details.

2.4 Composable security and Abstract Cryptography

As mentioned in the introduction, graph state verification protocols are usually used as sub-protocols in more complex protocols. They are meant to be reused many times, in a setting where a graph state is shared between n parties and some of the parties and/or the source may be dishonest. Hence, they do not only need to be secure for one run, but the security should also hold for an arbitrary number of repetition of the protocol. This level of security is called composable security, as opposed to game-based security where we study specific attack models against a protocol.

To prove composable security, one have to use a so called simulation-based framework where the security proofs are composable. In this paper, we will use the Abstract (or Constructive) Cryptography (AC) model, a top-down approach developed by U. Maurer and R. Renner [MR11b, MR16, Mau11]. This framework uses the concept of abstract systems to express cryptography as a resource theory. A cryptography protocol is viewed as the construction of some *ideal* resource $\mathcal{S}$ out of other *real* resources $\mathcal{R}$. Resources are box-like abstract systems with interfaces that are accessed by the parties. They represent any non-local operation such as communication channels, but also more involved functionalities to model for instance a coin tossing protocol. They can be composed in sequence or in parallel to create bigger resources.

Local operations—like the protocol run by a given party—are called *converters*. They are plugged into a single interface of a resource, changing the interaction of the resource with the outside world. A

converter π attached to a resource $\mathcal{R}$ creates a new resource that we write $\pi\mathcal{R}$. We usually call the converters π acting in the real world *protocols*, as they represent the local operations done by honest parties. In the ideal world, the *honest use of an ideal resource* is done through converters called *filters*, typically denoted $\perp$, that send the input expected from honest parties to the resource.

Finally, the distance between two resources is formalised through the notion of a distinguisher. It is an object that access all the interfaces of two given resources, such as $\pi\mathcal{R}$ and $\perp\mathcal{S}$, and tries to distinguish them by sending inputs and comparing the outputs. When considering statistical security, the distinguisher can be given unlimited computing power. On the other hand, computational security only allows distinguishers to have limited computing power. When any distinguisher accessing the interfaces of both resources cannot decide which of the two system is the ideal or the real one, we say that the resources are equivalent and write $\pi\mathcal{R} \approx \perp\mathcal{S}$.

Resources, converters and distinguishers are the building blocks of the AC theory. We refer the reader to [MR11b] for more details about the mathematical construction of the framework.

The secure construction of an ideal resource, that represents the ideal functionality that we want to achieve, from a concrete resource, that represents the actual realization of the protocol, is proven by showing a series of equivalences. In AC, a dishonest behaviour from a party is represented by unplugging the associated converter on the concrete resource, which means that this party is not following the protocol. This leaves some new interfaces accessible for a distinguisher on the concrete resource. To prove that the security of the protocol still holds, one has to find a converter called a *simulator* to plug into the ideal resource to make it indistinguishable from the dishonest concrete resource. The full security proofs thus consist in finding simulators for each possible subset of dishonest party. We give the security definition in AC below in Definition 2.1.

Definition 2.1 (Security in AC)**.** *let $\Pi = \{\pi_i\}_{i=1}^n$ be a protocol run by n parties using the concrete resource $\mathcal{R}$ and let $\mathcal{S}$ be an ideal resource with all the desired properties expected from the protocol. $\mathcal{R}$ and $\mathcal{S}$ have interfaces I. We say that Π* ***securely realizes*** *$\mathcal{S}$* ***out of*** *$\mathcal{R}$* ***within*** *ε or that Π ε-**realizes** $\mathcal{S}$ and write $\mathcal{R} \xrightarrow{(\Pi,\varepsilon)} \mathcal{S}$ if there exist simulators $\sigma = \{\sigma_i\}$ such that*[1]*:*

$$\forall H \subseteq I, \pi_H\mathcal{R} \approx_\varepsilon \sigma_{I\setminus H}\mathcal{S}\perp_H, \tag{3}$$

with $\forall H \subseteq I, \pi_H = \{\pi_i\}_{i\in H}$ and $\perp_H = \{\perp_i\}_{i\in H}$.

Remark 2.2. *Note that in this paper, we will equip the ideal resources with a communication channel that can forward any—possibly quantum—message between any party. This is only needed to allow simulators to perform non-local operations, and is filtered for honest parties. For a given subset of honest parties H, we will thus have to find only one global simulator $\sigma_{I\setminus H}$ such that the above equivalence relation holds.*

Since the AC framework is composable, any security proof proven within the framework is composable. This means all $\mathcal{R}, \mathcal{S}$ and $\mathcal{T}$ resources and π, ν converters (protocols) such that $\mathcal{R} \xrightarrow{\pi} \mathcal{S}$ and $\mathcal{S} \xrightarrow{\nu} \mathcal{T}$ we have that

$$\mathcal{R} \xrightarrow{\pi} \mathcal{S} \wedge \mathcal{S} \xrightarrow{\nu} \mathcal{T} \implies \mathcal{R} \xrightarrow{\nu\circ\pi} \mathcal{T}. \tag{4}$$

[1] Technically speaking [MR11a, Thm. 2], we should also define filters for $\mathcal{R}$ possibly restricting the access to $\mathcal{R}$ for honest participants. For simplicity, and without loss of generality, we will often assume that the filters for $\mathcal{R}$ are trivially forwarding their inputs, and we integrate the original filters in the parties. This way, proving the correctness of a protocol using $\mathcal{R}$ can be simplified as $\pi_H\mathcal{R} \approx \perp_H\mathcal{S}\sigma_M$ instead of $\pi_H\perp_H^{\mathcal{R}}\mathcal{R} \approx \perp_H\mathcal{S}\sigma_M$.

3 Composable security of graph state verification

3.1 Definition of the ideal verification resources

In this section, we present the ideal functionalities that we will use to prove the security of generic graph state verification protocols.

Ideal resource. Note than in this article we will consider two functionalities:

- $\mathcal{V}^f_{|G\rangle}$ (Resource 3.1) is the functionality that we consider in this section: on the one side, we can show that any secure graph state verification algorithm realizes this functionality, without any modification to the protocol. However, this functionality is less natural than what one would expect, as it also allows malicious adversaries to apply some malicious corrections to the graph state given to honest adversaries. The function f acts as a safeguard to only allow some corrections to be performed by the adversary. More precisely, f must be a function taking as input a subset of corrupted party M and (x, z), a list of X and Z corrections to apply on the qubits sent to honest parties, and must output either $\top$ if this correction is allowed or $\perp$ otherwise. We discuss later in Remark 3.4 the motivations behind this verification.
- $\mathcal{V}_{|G\rangle}$ (Resource 4.1), on the other hand, is much simpler and closer to what one would expect as it simply sends $|G\rangle$ to all parties (unless the malicious parties abort), but we cannot directly show that all protocols realize this functionality: we need to apply an additional step where all parties apply a random stabilizer on their part of the state.

Since both resources have pro and cons, we study them separately in the following two sections, starting from $\mathcal{V}^f_{|G\rangle}$. $\mathcal{V}^f_{|G\rangle}$, that we describe formally in Resource 3.1 and informally in Fig. 1, is an abstract system outputting all qubits of a given graph state $|G\rangle$ or an abort signal, with interfaces allowing to model possible dishonest behaviors from the parties and the source. Note that $\mathcal{V}^f_{|G\rangle}$ is not a functionality allowing to create any graph state, but for any graph state $|G\rangle$, we can construct an ideal resource $\mathcal{V}^f_{|G\rangle}$ outputting $|G\rangle$.

The $\mathcal{V}^f_{|G\rangle}$ resource has $n+1$ interfaces, one for each party and one for the source. For each party i, the value $c_i \in \{0, 1, \perp\}$ indicates whether the party is honest or dishonest. To an honest party ($c_i = 0$), it will output either an Abort signal or a qubit from the graph state $|G\rangle$. To a dishonest party ($c_i = 1$ or $\perp$), it will output the corresponding qubits of $|G\rangle$ and then wait for corrections (a_i, b_i). For readability, in Fig. 1, we show on the left the input and output corresponding to honest uses of the resource from the parties, while the bottom input and output correspond to dishonest behavior. It should however be kept in mind that only one input/output interface is accessible to each party. The resource $\mathcal{V}^f_{|G\rangle}$ is equipped with a function f that outputs a Boolean stating whether the corrections proposed by the dishonest parties have the correct form. The details of how the resource works is given below in Protocol 1, and the details of f will be explicit in the security proof.

In Abstract Cryptography, we typically model the *honest use of an ideal resource* by adding special converters called *filters* that send the input expected from honest parties to the resource. In our case, an honest use of the $\mathcal{V}^f_{|G\rangle}$ resource corresponds to the source sending $c_S = \top$ and each party i sending $c_i = 0$ to the resource. We thus define the following: the filter $\perp_S$ that corresponds to an honest source sending $c_S = \top$ and, for $i \in [n]$, the filter $\perp_i$ that corresponds to an honest party sending $c_i = 0$ to $\mathcal{V}^f_{|G\rangle}$. For the sake of simplicity, we will write $\perp_{[n]}$ the filter corresponding to the parallel composition of $\perp_i$ for all $i \in [n]$. An honest use of the ideal resource is thus represented by the so-called filtered resource, that we show in Fig. 2.

Resource 3.1 Ideal resource $\mathcal{V}^f_{|G\rangle}$

1. Receive from the source's interface $c_{\mathtt{S}} \in \{\top, \bot\}$
2. If $c_{\mathtt{S}} = \bot$, abort and send $\bot$ to all parties. Otherwise it continues.
3. Create $|G\rangle$.
4. Receive $\{c_i\} \in \{0, 1, \bot\}^n$ from each party. If any $c_i = \bot$ send $\bot$ to all parties and abort. Otherwise, let $M = \{c_i \mid c_i = 1\}$ be the set of malicious parties, $H = [n] \setminus M$ the honest parties, $m = |M|$ the number of malicious parties and $h = |H|$ the number of honest parties.
5. For each $i \in M$, send the i-th qubit of $|G\rangle$ to interface i.
6. Receive from each malicious party $i \in M$ corrections $(a_i, b_i) \in (\{0,1\}^h)^2$, and define $x = \oplus_{i\in M} a_i = \{x_i\}_{i\in H}$ and $z = \oplus_{i\in M} a_i = \{z_i\}_{i\in H}$.
7. If there is a least one malicious party, check if $f_G(M, x, z) = \top$ (to check that corrections are well formed), if not sent $\bot$ to all parties and abort.
8. If ok, Apply $\mathbf{Z}^{z_1} \otimes ... \otimes \mathbf{Z}^{z_h}$ and $\mathbf{X}^{x_1} \otimes ... \otimes \mathbf{X}^{x_h}$ to the remaining qubits of $|G\rangle$.
9. Send these qubit to the parties.

Additionally, we include in $\mathcal{V}^f_{|G\rangle}$ an additional communication channel $\mathcal{C}$ that can forward any—possibly quantum—message between any party (see Remark 2.2). We also define naturally the filter $\bot_{\mathtt{S}}$ as the converter that sends $\top$, and for any $i \in [n]$, we define $\bot_i$ as the converter that sends $c_i = 0$ and forwards any message sent or received by the ideal function functionality (excluding messages sent on $\mathcal{C}$ that are just blocked).

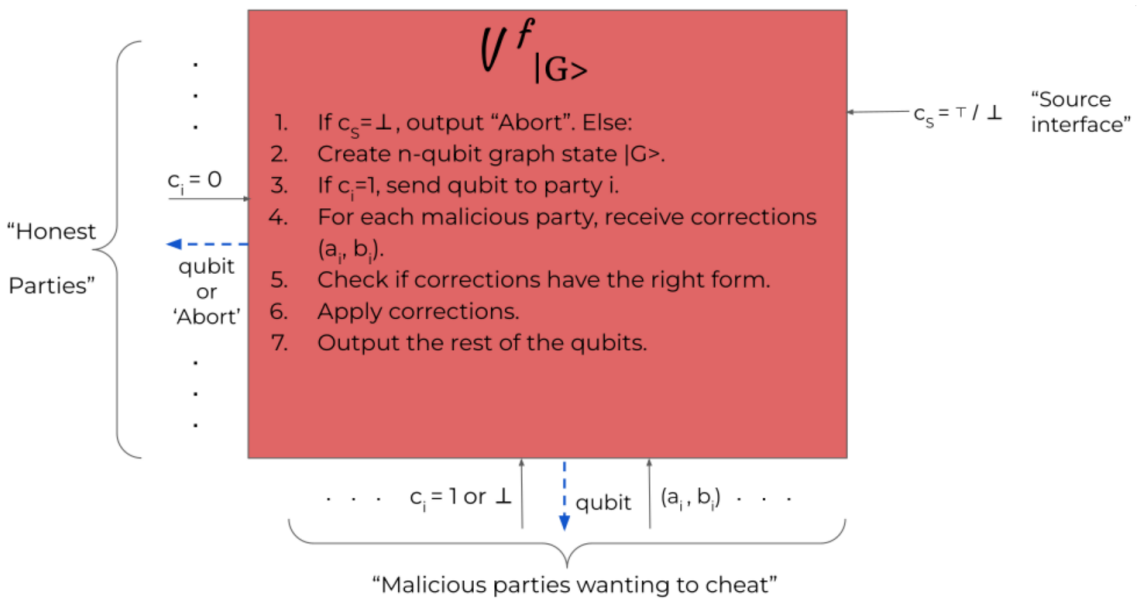


Figure 1: Informal presentation of the ideal resource $\mathcal{V}^f_{|G\rangle}$ for verified graph state sharing.

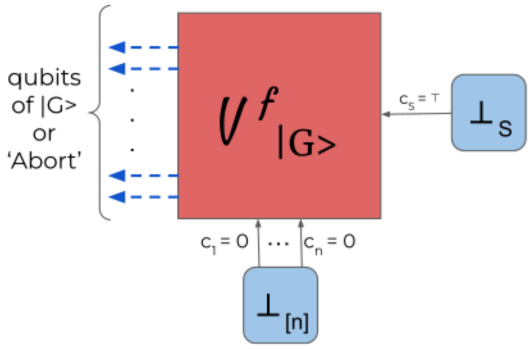


Figure 2: Ideal filtered resource. The filter $\perp_S$ corresponds to an honest source inputting $c_S = \top$ and, for $i \in [n]$ the filter $\perp_i$ corresponds to an honest party sending $c_i = 0$ to $\mathcal{V}^f_{|G\rangle}$

3.2 Definition of the concrete verification protocols

Generally, protocols to verify the preparation of graph states consist of the following steps: first, quantum states are shared between n parties, one qubit of each state per party, then, they test a random selection among these states and if the checks are positive, they keep the others for later use. The tests usually consist in local operations and measurements, classical communication and multiparty computation that outputs a bit indicating if the state is far away the desired state reduced to the honest parties, for a certain distance (usually the trace distance). By randomising which states they test and which one they use, the parties prevent a malicious source from sending the desired states. Verification protocols can also be done sequentially, by asking the source to send quantum states one after the other and randomly choosing the ones that they test and the ones that they use.

Verification protocols provide a bound on the probability that, given the protocol has not aborted after a certain number of tests, the final reduced state that is used for computation or communication by

the honest parties is close to the desired graph state reduced to the honest parties. This bound depends on the number of parties, the number of tests, the number of dishonest parties and the maximum distance that is accepted with the desired graph state.

Definition of graph state verification protocols. The literature uses different (but mostly equivalent) security definitions when considering graph state verification protocols. Note that the security of these protocols is never expressed in the composable AC framework. The main contribution of this article is actually to prove that any protocol fulfilling the security definition of Definition 3.1 below is also composably secure.

Definition 3.1 (Graph state verification protocol)**.** *Let $\Pi = \{\pi_i\}_{i\in[n]} \cup \{\pi_{\mathtt{S}}\}$ be a protocol between n parties and a source, interacting through a resource $\mathcal{R}$ in charge of modeling, for instance, the communication channels between all parties. We will say that Π is a* ε-graph state verification protocol *if the following properties are respected:*

- ***Correctness**: if all parties are honest, they output a state negligibly close (in trace distance) to $|G\rangle$, i.e. $\pi_{[n]}\mathcal{R}\pi_{\mathtt{S}}$ outputs ρ such that $\mathrm{TD}(\rho, |G\rangle) \geq 1 - \mathsf{negl}(\lambda)$.*
- ***Security**: for any set of honest parties $H \subseteq [n] \cup \{\mathtt{S}\}$, the honest parties output their state at the same time[2], each party outputting either a special symbol $|\perp\rangle$ if they aborted, or a quantum state otherwise[3]. Moreover, when considering any adversary[4] $\mathcal{A}$ corrupting parties in $[n] \setminus H$, there exists $p \in [0,1]$ such that:*

$$F(\rho, \sigma) \geq 1 - \varepsilon(\lambda) \tag{5}$$

where F is the fidelity (we use the definition of [NC10], sometimes called the square root fidelity), $\rho := \mathbb{E}\left[\mathrm{Tr}_{[n]\setminus H}\,\rho_i \;\middle|\; \rho_i \leftarrow \pi_H \mathcal{R}\mathcal{A}\right]$ is the averaged state obtained by the honest parties at the end of the protocol, where we average over all randomness involved in $\mathcal{A}$ and in the whole protocol[5] and $\sigma := p\,\mathrm{Tr}_{[n]\setminus H}(|G\rangle\langle G|) + (1-p)\,|\perp^{|H|}\rangle\langle\perp^{|H|}|$ denotes the mixture where all honest parties either abort at the same time with probability $1-p$ or output a qubit that is part of $|G\rangle$.

Concrete resource In this work, we model all graph state verification protocols in the same way: we abstract all the resources required to perform these protocols into one resource $\mathcal{R}$ and the local operations done by each parties into converters $\{\pi_i\}_{i=1}^n$. For instance, $\mathcal{R}$ can contain the quantum channel from the source to each party, some authenticated classical channels used to communicate between the parties, a coin flipping resource or a multiparty computation resource that may involve shared randomness resources etc. The converters $\{\pi_i\}_{i=1}^n$, that we will write $\pi_{[n]}$, correspond to the protocol followed by the honest parties and often consist in applying some quantum operations on their qubit, measuring them and using the output in the multiparty computation that tests the state. Finally the source's protocol $\pi_{\mathtt{S}}$ generally consists either in sending a certain number of copies of a desired state or in sending them one by one. We illustrate in Fig. 3 the concrete resource $\pi_{[n]}\mathcal{R}\pi_{\mathtt{S}}$ corresponding to an honest run of the protocol, which simply outputs a state that is close to a fixed graph state $|G\rangle$.

[2]This can for instance be done using a broadcast channel, which is anyway implicitly needed in most existing works.

[3]Note that this implies that the Hilbert space is spanned by $\{|0\rangle, |1\rangle, |\perp\rangle\}$, where $|\perp\rangle$ is orthogonal to the other two states. This is quite practical as this way we can only maintain n registers instead of $2n$, and we do not need to worry about the content of the other register when the abort register contains an abort. Note that otherwise, this is mostly equivalent as we can test if a party aborted by simply measuring $(|\perp\rangle\langle\perp|, I - |\perp\rangle\langle\perp|)$, without disturbing the state if it is only spanning $(|0\rangle, |1\rangle)$.

[4]$\mathcal{A}$ might be bounded or unbounded depending on the assumptions on the protocol.

[5]We can also equivalently purify $\mathcal{A}$ and the protocols using the Stinespring dilation, and simply trace-out all the registers except for the output register owned by the honest party. The Stinespring dilation allows us to postpone any measurement by essentially replacing them with a **CNOT** on an auxiliary qubit, and to sample randomness by creating a $|+\rangle$ state measured using the postponed measurement that we just described.

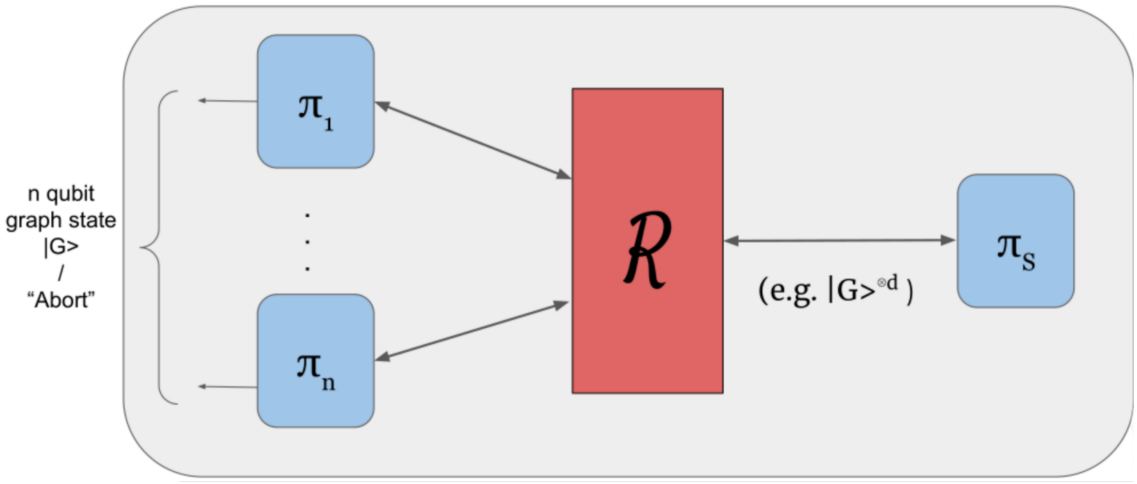


Figure 3: Concrete resource for generic graph state verification protocols

3.3 Security proof

In this section we prove our main result, namely the composable security of any graph state verification protocol. Following the Security Definition 2.1, we will prove the following theorem in the rest of this section:

Theorem 3.2. *Let $|G\rangle$ be an arbitrary graph state, and let Π be an ε-graph state verification protocol according to Definition 3.1. Then, Π $(2\sqrt{2\varepsilon - \varepsilon^2})$-realizes the functionality $\mathcal{V}^f_{|G\rangle}$, where the allowed corrections f are defined as follows: let U, r, R be defined like in Lemma A.11, then we define $f(M, x, y)$ as the function that outputs $\top$ iff the last $|n| - r$ vectors of Ux are 0's, and if $(U^T)^{-1}(z \oplus G_H x) = \begin{bmatrix} b \\ R^T b \end{bmatrix}$ for some arbitrary vector $b \in \{0,1\}^r$.*

3.3.1 Correctness

Proof. The first step is to prove the correctness of the protocol when all parties are honest, i.e. we need to show that $\pi_{[n]}\mathcal{R}\pi_{\mathsf{S}} \approx \perp_{[n]}\mathcal{V}^f_{|G\rangle}\perp_{\mathsf{S}}$. This is also pictured in Fig. 4, where on the left-hand side we represented the filtered version of the $\mathcal{V}^f_{|G\rangle}$ resource, simply outputting the final—expected—state, and corresponding to a honest use of the resource.

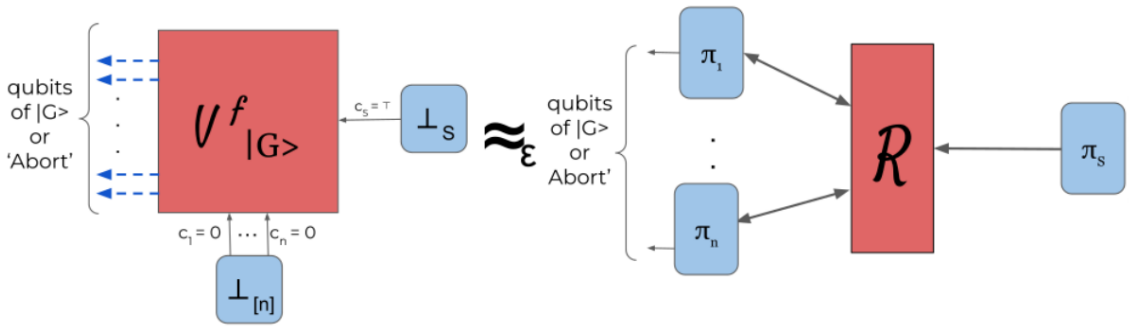


Figure 4: Correctness of the protocol: $\perp_{[n]}\mathcal{V}^f_{|G\rangle}\perp_{\mathsf{S}} \approx \pi_{[n]}\mathcal{R}\pi_S$

This is actually a direct consequence of the correctness of the protocol. First, we can easily see that since the filters disable any malicious behavior, $\perp_{[n]}\mathcal{V}^f_{|G\rangle}\perp_{\mathtt{S}}$ is the system that outputs $|G\rangle$ with probability 1. Moreover, due to the correctness property of Π, $\pi_{[n]}\mathcal{R}\pi_{\mathtt{S}}$ outputs a state indistinguishable from $|G\rangle$. Hence, any distinguisher trying to distinguish between the two resource only gets indistinguishable copies of $|G\rangle$, i.e. $\pi_{[n]}\mathcal{R}\pi_{\mathtt{S}} \approx \perp_{[n]}\mathcal{V}^f_{|G\rangle}\perp_{\mathtt{S}}$. This ends the correctness proof. □

3.3.2 Security

The next step in the security proof, (see Definition 2.1), is to study the case of dishonest parties tampering with the verification protocol. This means that some subset $M \in [n]\cup\{\mathtt{S}\}$ of the parties, possibly including the source, are no longer following their local protocols π_i. The concrete resource representing the real protocol becomes $\pi_H\mathcal{R}$ where $H \subseteq [n]\cup\{\mathtt{S}\}$ are the honest parties.

On the ideal resource, the malicious behavior of some subset M of the parties is represented by removing the filters $\{\perp_i\}_{i\in M}$, letting the adversaries access interfaces of the ideal resource $\perp_H\mathcal{V}^f_{|G\rangle}$. We need to find a simulator σ_M such that there exists an ϵ such that $\pi_H\mathcal{R} \approx_\epsilon \perp_H\mathcal{V}^f_{|G\rangle}\sigma_M$. We show the representation in AC in Fig. 5.

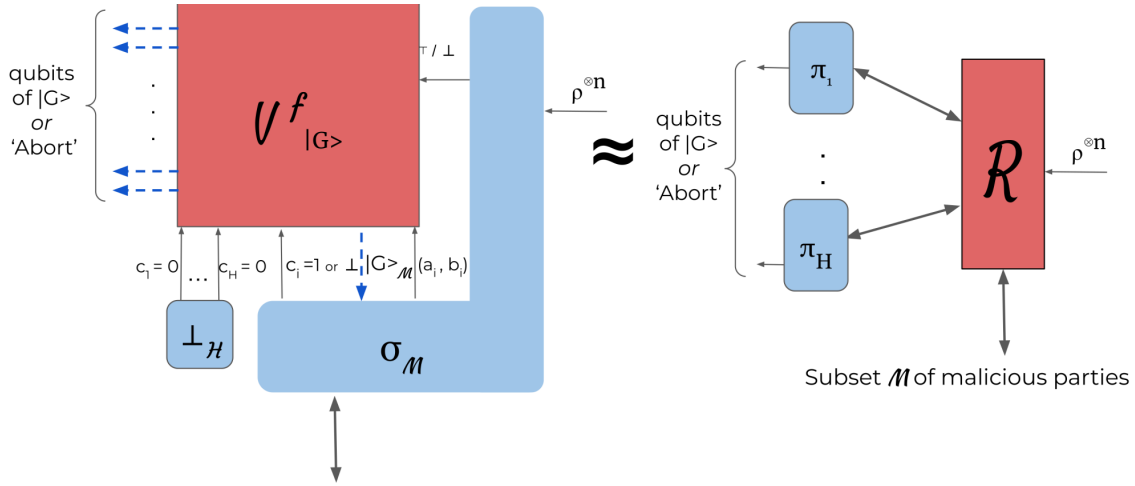
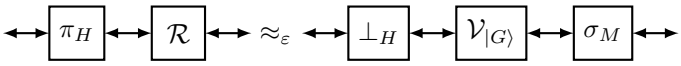


Figure 5: General case of the security proof: we need to find σ_M such that $\pi_H\mathcal{R} \approx_\epsilon \perp_H\mathcal{V}^f_{|G\rangle}\sigma_M$

In order to construct such a simulator, and thus to prove that all graph state verification protocols are composable, we first study a property of graph states that we call "mergeable".

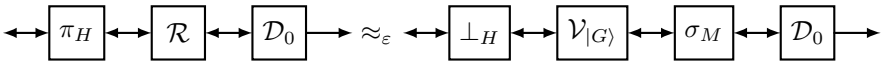
Mergeable states. In order to motivate this property, let us try to prove that a given graph state verification protocol $\Pi = (\pi_1, \ldots, \pi_n, \pi_{\mathtt{S}})$ is composably secure, where $\pi_{\mathtt{S}}$ is the protocol followed by the source while π_i's are the protocols followed by other parties. As explained above, this is done by finding a simulator to emulate on the ideal resource the interfaces of the concrete resource left open when removing the converters π_i corresponding to the dishonest parties. Let us try naively to find such a simulator. If H is the set of honest parties and $M := \bar{H}$ is the set of malicious parties, possibly including the source, we want to find a simulator σ_M such that $\pi_H\mathcal{R} \approx_\varepsilon \perp_H\mathcal{V}_{|G\rangle}\sigma_M$:



(6)

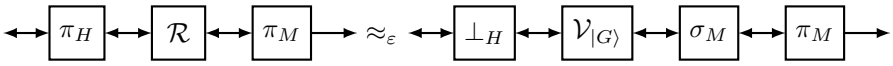
where we represented the interfaces of honest parties on the left and the interface of malicious parties on the right, and where $\perp_H$ is the filter blocking the honest interfaces of the ideal resource.

For the sake of explanation, let us first try to find a simulator that works when the distinguisher is first running the honest protocol π_M (let us write this part of the distinguisher $\mathcal{D}_0$): in that case, we can rewrite Eq. (6), which gives us:



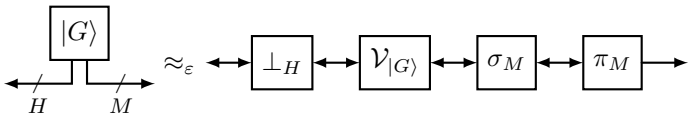
(7)

i.e., after replacing the definition of this first naive distinguisher:



(8)

Due to the correctness of Π, in the real world both parties will share $|G\rangle$, i.e. we can simplify the LHS with:

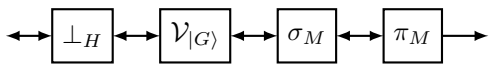


(9)

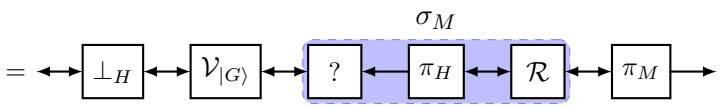
Since we want our simulator to work for any protocol Π, the most natural thing to do is to let the simulator start with simulating $\pi_H\mathcal{R}$ directly in a black-box manner otherwise we would not even know what to send to the distinguisher. The simulator hence forwards all messages received on the outer interface to the appropriate honest party via $\mathcal{R}$, and all messages from honest parties to the outer interface. At the end of this interaction, the simulator will obtain a state outputted by the honest parties: if one party aborted, the simulator can tell the ideal functionality to abort but this should never occur in this simplified analysis where the distinguisher is running the honest protocol. If no party aborted, and if Π is secure, then:

- we should get from π_H a state close to $|G\rangle^H$,
- and on the other hand, the simulator will receive from the ideal functionality $\mathcal{V}_{|G\rangle}$ a state $|G\rangle^M$.

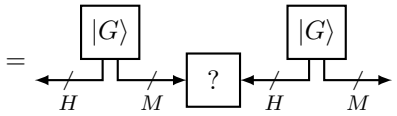
By the correctness of π and the definition of $\mathcal{V}_{|G\rangle}$, we can therefore rewrite the RHS of Eq. (9) as follows, where $\boxed{?}$ represents a yet unknown operation run by the simulator that we need to determine:



(10)

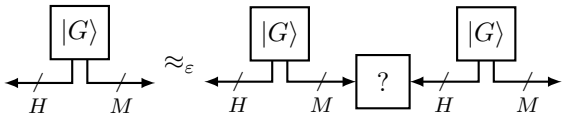


(11)



(12)

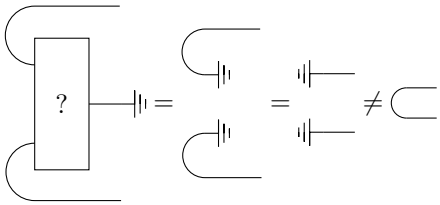
We can inject this back into Eq. (9) to get:



(13)

We should therefore search for a "merging" operation that can combine two shared state $|G\rangle$ into a single copy of $|G\rangle$ while working *only* on (different) parts of two copies of $|G\rangle$. Unfortunately, it is easy to see that it is impossible to obtain such a merging map. For instance, if we take $|G\rangle$ to be a Bell pair, one of

the simplest graph state, then after tracing out the registers of $\boxed{?}$ (that cannot signal any measurement outcome to the outside world) we obtain two un-entangled states, which cannot possibly be equal to a single entangled Bell pair. This can be seen for instance diagrammatically using the formalism described in [CK17], or using non-signaling as formalised in Theorem 3.3:



(14)

This is not surprising since, for instance, in quantum teleportation after the Bell measurement Bob needs to apply additional corrections to recover the original state. We formalise now this statement:

Theorem 3.3 (Impossibility of black-box realization of $\mathcal{V}_{|G\rangle}$)**.** *There exists some*[6] *graph states* $|G\rangle$ *such that for any graph state verification protocol* $\Pi \coloneqq \{\pi_i\}_{i\in[n]\cup\mathsf{S}}$ *using a resource* $\mathcal{R}$ *and producing* $|G\rangle$ *(i.e.* $\{\pi_i\}_{i\in[n]\cup\mathsf{S}}\mathcal{R}$ *outputs* $|G\rangle$ *shared among the n parties), it is* impossible *to prove that* Π *is* ε*-realizing* $\mathcal{V}_{|G\rangle}$ *(Resource 3.1) for any* $\varepsilon < 1/2$ *if the simulator is black-box*[7]*, in the sense that the simulator interacts with the interface controlled by the environment by running* $\{\pi_i\}_{i\in H}\mathcal{R}$*, forwarding all messages between the malicious interfaces of* $\mathcal{R}$ *and the environment.*

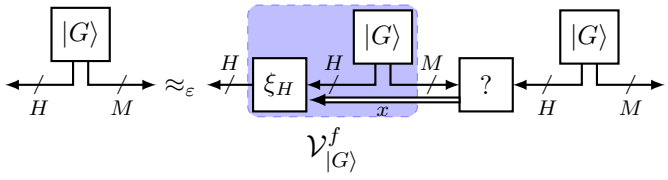
See proof in Appendix B.1.

We are therefore left with two options:

- either we change the functionality,
- or we change the protocol.

We present both approaches in this article. We will focus in this section on the first approach, while the second approach will be seen in Section 4, building on the results introduced here.

The previous impossibility result suggests a first modification: Let the simulator be allowed to communicate the output x of their measurements to the ideal functionality. This in turns let the functionality apply these corrections $\xi_H(x)$ to the honest part of the graph state. This brings us to the following picture, building on Eq. (13):



(15)

Remark 3.4. *It is important to allow only certain, harmless, corrections, by letting the functionality check that the corrections* x *are valid, after verifying that* $f(x) = \top$*. Indeed, allowing arbitrary corrections would allow the adversary to perform attacks that might be impossible to perform with only access to*

[6] Actually most graph states have this property as soon as they are not separable.

[7] We call it black-box since the definition of the simulator is mostly independent of the protocol, as it can only execute Π without having access to its code. This definition of black-box simulator is relatively generic: if the simulator does not know the definition of the protocol Π, it can basically only forward the messages of the protocol to the distinguisher. For instance, if all exchanged messages are signed with a key unknown to the simulator (but which is part of the public description of the protocol), the only way to communicate with the distinguisher is to forward the messages sent by running $\{\pi_i\}_{i\in H}\mathcal{R}$. We could formalize this by giving an even more generic definition of black-box simulator that is not explicitly asked to forward the messages of $\{\pi_i\}_{i\in H}\mathcal{R}$ to the distinguisher, but this would obfuscate our proof without clear benefits.

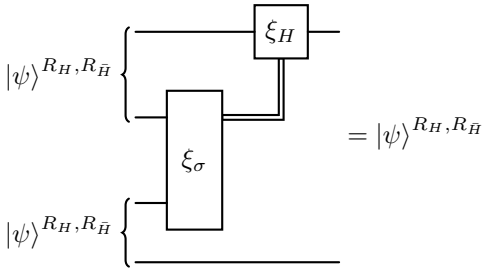
$|G\rangle^M$. *For instance, if we allow arbitrary corrections on a GHZ state* $\frac{1}{\sqrt{2}}(|0\dots0\rangle + |1\dots1\rangle)$*, then nothing prevents an adversary from flipping the bit of the honest parties in a different way, for instance to produce* $\frac{1}{\sqrt{2}}(|10\dots0\rangle + |01\dots1\rangle)$*, where the honest parties get the first two qubits. This would be disastrous, for instance if we build a coin tossing protocol from this GHZ state by measuring the first two qubits in the computational basis, since the first two parties would get different outcomes, which is impossible to obtain from a normal GHZ. Note that it might be hard to justify why a correction* x *is harmless as it might depend on the protocol that we target. Protocols that expect a graph state* $|G\rangle$ *should in theory reprove that they are secure when using the functionality* $\mathcal{V}^f_{|G\rangle}$*, or use, instead, the result that we present in Section 4. However, we believe that the set of corrections we obtain is harmless, since it is basically a subset of the stabilizers of the graph state. Therefore, intuitively, any correction* x *such that* $f(x) = \top$ *can be done by the adversary by applying a stabilizer on his side, which will automatically propagate some corrections to the honest players. This property is actually formalized later in order to obtain our second result in Section 4.*

Informally speaking, a state is *mergeable* if the following holds: if this state is shared between two parties Alice and Bob, and if another copy of this state is shared between Bob and Charlie, then is is be possible to obtain a single copy of this state between Alice and Charlie, under the constraint that Charlie should not do any operation, that Bob should perform an arbitrary measurement to obtain an outcome m, and that Alice only does an operation that only depends on m. In a sense, this can be seen as a generalization of entanglement swapping to arbitrary states, with additional constraints on the set of allowed operations. Said differently, we will say that a state $|G\rangle$ is mergeable with respect to f and ξ_H if we can find —[?]— (called ξ_σ from now) such that Eq. (15) is true, where $f(x) = \top$. Since this must be true for any set of corrupted party, the function f and ξ_H will be different for any subset H of honest parties. More formally:

Definition 3.5 (Mergeable states)**.** *A state* $|\psi\rangle^{R_1,\dots,R_n}$ *on* n *registers* $\{R_i\}_{i\in[n]}$ *is said to be* mergeable *with respect to a function* $f\colon \mathcal{P}([n]) \times \mathcal{M}$ *(taking as input a set of honest party and a measurement outcome), and a collection of quantum maps* $\{\xi_H\}_{H\subseteq[n]}$ *(taking as input the registers* R_i *for* $i \in H$*, together with an additional classical input, and outputting the same* R_i*'s registers) if for any subset* $H \subseteq [n]$ *of registers, there exists a quantum map* ξ_σ *(taking inputs from the registers* $R_{\bar{H}}$ *of a first state, and* R_H *of a second state, and outputting a single classical value in* $\mathcal{M}$ *such that* $f(H, x) = \top$*) and* ξ_H *(taking the register* R_H *of the first state and the classical output of* ξ_σ*, and outputting a quantum register), such that:*

$$(\xi_H \otimes I_{n-|H|})(I_{|H|} \otimes \xi_\sigma \otimes I_{n-|H|})(|\psi\rangle^{R_H,R_{\bar{H}}} \otimes |\psi\rangle^{R_H,R_{\bar{H}}}) = |\psi\rangle^{R_H,R_{\bar{H}}} \tag{16}$$

Note that this is always trivially possible if H *is empty or equal to* $[n]$*. In picture:*



(17)

We show now that any graph state is mergeable. Note that during a first reading, it might help to start with the simpler construction of Corollary 3.7 that focuses only on GHZ states.

Theorem 3.6 (Any graph state is mergeable)**.** *For any graph* $G = (V, E)$*,* $|G\rangle$ *is mergeable (Definition 3.5) with respect to the maps* $\{\xi_H\}_{H\subseteq[n]}$ *that take two lists of* X *and* Z *corrections* $(x, z) \in (\mathbb{Z}_2^{|H|})^2$ *and applies* $\mathbf{X}^x\mathbf{Z}^z$ *on the input qubits.*

The merge procedure is described diagrammatically in Fig. 6, and for completeness we reformulate it here. Let $n = |V|$, H and M be any partition of V. For simplicity, we assume that we reorder elements of V to have elements of H ordered before elements of M. Let Γ be the biadjency graph between H and M (cf. Definition A.4). Then, we define ξ_σ as follows (cf. illustration Fig. 6), where the i-th qubit of $|G\rangle$ belongs to register H (resp. M) iff $i \in H$ (resp. M).

- *It applies $\wedge\mathbf{Z}$ gates on any pair (i, j) of qubits of register H iff $(i, j) \in G_H$ and similarly it applies $\wedge\mathbf{Z}$ gates on any pair (i, j) of qubits of register M iff $(i, j) \in G_M$.*
- *It applies Hadamard gates on all qubits of register M.*
- *It computes U, V, r and R according to Lemma A.11 and applies the unitary $|x\rangle \mapsto \left|V^{-1}x\right\rangle$ on register M and $|x\rangle \mapsto |Ux\rangle$ on register H. This is always possible since U and V are invertible. We propose moreover in Lemma A.12 a way to implement them more efficiently, without auxiliary qubits and using only* **CNOT** *and swap operations.*
- *It performs r Bell measurements (projection on one of the four Bell states) between the first r qubits of each register. The Bell measurements are between the i-th qubit of register M with the i-th qubit of register H, where a measurement outcome $(b_i, c_i) \in \{0, 1\}^2$ means that the i-th pair was projected on the Bell state $|0c_i\rangle + (-1)^{b_i} |1\bar{c_i}\rangle$. The outcomes are gathered into two vector $b = (b_i)_{i\in[r]}$ and $c = (c_i)_{i\in[r]}$.*
- *It performs a measurement in the $\{H |a_i\rangle\}_{a_i\in\{0,1\}}$ basis on the $|M| - r$ remaining qubits of register M (the outcomes are gathered into a vector a), and a measurement in the computational basis $\{|d_i\rangle\}_{d_i\in\{0,1\}}$ on the $|H| - r$ remaining qubits of register H (the outcomes are gathered into a vector d).*
- *It computes $x \coloneqq U^{-1} \begin{bmatrix} c \oplus Rd \\ \mathbf{0} \end{bmatrix}$, and $z \coloneqq \left(U^T \begin{bmatrix} b \\ R^T b \end{bmatrix}\right) \oplus Gx$ where Gx is the set of neighbours of x as defined in Lemma A.7, and returns the corrections (x, z).*

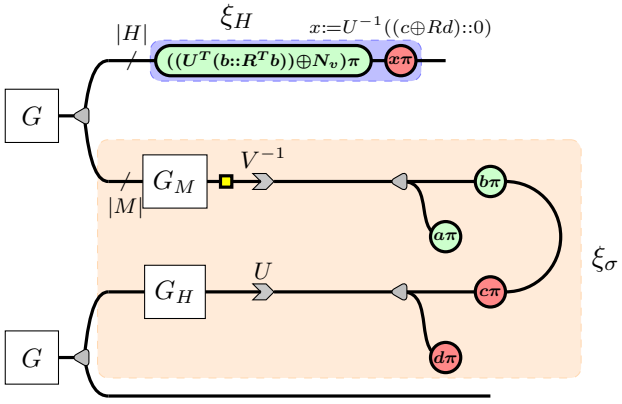


Figure 6: Representation in ZX-calculus of the procedure to merge two copies of a graph state into a single copy. The orange area denotes the merging map ξ_σ while the blue one represents the corrections ξ_H to apply. Note that H and M form a partition of the vertices of G, and that G_H, G_M, Γ, U, V and R are defined like in Definition A.4 and Lemma A.11.

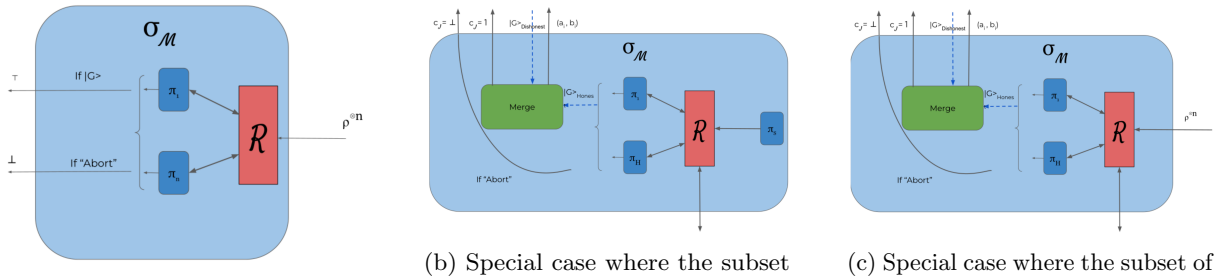
Note that this lemma is at the heart of our construction. Since the proof is quite technical and heavily relies on scalable ZX-calculs, we defer the full proof to Appendix B.1.

Since the above theorem is true for any graph state, it is also true for GHZ states. However, the merging operation can be significantly simplified in that setting:

Corollary 3.7 (GHZ states are mergeable)**.** *Any* GHZ *state of size* $|n|$ *(each qubit being a separate register) is mergeable (Definition 3.5) with respect to the collection of quantum maps* $\{\xi_H\}_{H\subseteq[n]}$*, where* ξ_H *takes two bits* $(x,z)\in\{0,1\}^2$ *as input, applies* $\mathbf{Z}^z$ *of the first qubit, and* $\mathbf{X}^x$ *on all its input qubits (if* $|H|$ *is empty, it does not do anything).*

See proof in Appendix B.1.

Security proof of Theorem 3.2. Building on this corollary, we prove now the security part of Theorem 3.2, schematised in Fig. 5. The simulators that we will define are informally drawn in Fig. 7.



(a) Special case where the subset of corrupted parties M contains only the source (i.e. $M=\{\mathtt{S}\}$).

(b) Special case where the subset of corrupted parties M contains only some parties (excluding the source).

(c) Special case where the subset of corrupted parties M contains the source and a coalition of dishonest parties.

Figure 7: Simulators for (a) a dishonest source, (b) a subset of dishonest parties and (c) a coalition of dishonest parties and source. Note that even if we describe three simulators depending on the set of corrupted resources for clarity, in the proof we directly describe the generic case where the set M of corrupted parties can be arbitrary.

Lemma 3.8 (Security proof of Theorem 3.2)**.** *Let* $M\subseteq[n]\cup\{\mathtt{S}\}$ *be an arbitrary subset of corrupted parties, and* $H=[n]\cup\{\mathtt{S}\}\setminus M$ *be the honest parties. Then there exists a simulator* σ_M *such that* $\pi_H\mathcal{R}\approx\perp_H\mathcal{V}^f_{|G\rangle}\sigma_M$ *for each possible subset* M*.*

Proof sketch. This proof is relatively technical and is therefore deferred to appendix. Yet, the informal idea is to use the fact that the state is mergeable. More precisely, the simulator can simulate the protocol run by honest parties, forwarding all messages from/to the distinguisher, to get the state outputted by the honest parties. If no party aborted, then the state should be close enough to $|G\rangle^H$ (part of the technicality is to write this properly since part of this state is owned by the adversary). In that case, the simulator will then get from $\mathcal{V}^f_{|G\rangle}$ another copy of $|G\rangle^M$, and use the fact that this state is mergeable in order to merge these two copies of $|G\rangle$ into a single state close to $|G\rangle$ up to some corrections, shared between the distinguisher and the functionality. The merging operation will provide these corrections to the simulator, that can forward them to the ideal resource. The resource will then apply these corrections to ensure that the shared state is now close to $|G\rangle$, before outputting its part of $|G\rangle$ to the distinguisher. □

See full proof in Appendix B.1.

Note that while we focus in this proof only on graph states, the result in this section also easily extend to any mergeable state.

4 Realizing the ideal resource without corrections

We showed in Theorem 3.2 that all verification protocols, provided some basic security properties, realize without any change the resource $\mathcal{V}^f_{|G\rangle}$. However, this resource does allow the adversary to apply some

corrections on the state of the honest parties. While we believe that such corrections are harmless for the security as they can already, to a certain degree, be created by applying some operations on the state owned by the malicious party, one might prefer a "cleaner" interface for the resource, at the cost of having a slightly more involved protocol. In this section, we show that any protocol realizing $\mathcal{V}^f_{|G\rangle}$ can be turned into a new protocol that realizes a new resource $\mathcal{V}_{|G\rangle}$ (note that the dependency on f is gone).

The modification consists in applying a random stabilizer of G on their share of the graph state at the end of the protocol. We abstract the sampling and distribution of this stabilizer in a separate, additional, resource. In practice, this operation can for instance be done via a coin flipping protocol.

We formally describe the resource $\mathcal{V}_{|G\rangle}$ in Resource 4.1. Apart from allowing malicious adversaries to abort and to get their share of the quantum state before honest parties, it simply sends $|G\rangle$ to all parties. We also define the filters $\perp_i$ for each party i that sends $c_i = \top$ and we show an informal representation of the filtered resource $\perp_{[n]\cup\mathsf{S}}\mathcal{V}_{|G\rangle}$ in Fig. 8.

Resource 4.1 Ideal resource $\mathcal{V}_{|G\rangle}$

---

1. Create a quantum state $|G\rangle$.
2. Receive for each party i a bit $c_i \in \{\top, \perp\}$. If any $c_i = \perp$, send the i-th qubit of $|G\rangle$ to party i, and wait for another abort bit $c'_i \in \{\top, \perp\}$: if any $c'_i = \perp$, abort by sending $\perp$ on all interfaces; otherwise, for each i such that $c_i = \top$, send the i-th qubit of $|G\rangle$ to party i.

Additionally, we include in $\mathcal{V}_{|G\rangle}$ an additional communication channel $\mathcal{C}$ that can forward any—possibly quantum—message between any party (see Remark 2.2).

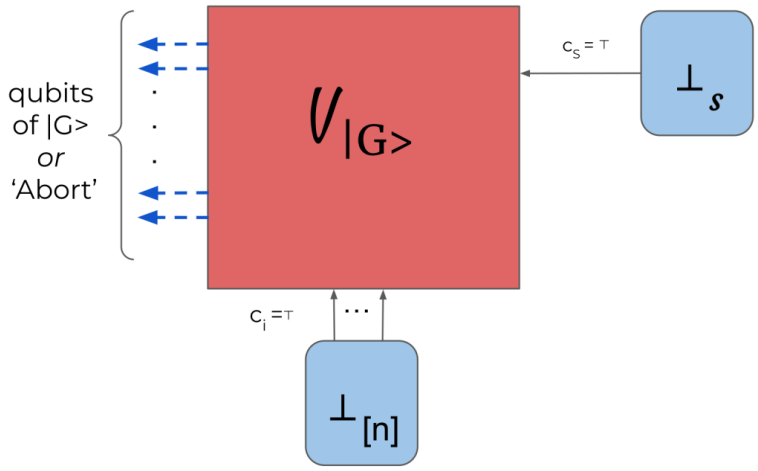


Figure 8: Ideal filtered resource $\pi_{[n]\cup\mathsf{S}}\mathcal{V}_{|G\rangle}$ representing a more "clean" ideal graph state verification resource than in the previous section.

We define in Resource 4.2 the ideal functionality that samples a random x, abstracting a common randomness generation resource. Note that one might want a simpler version of Resource 4.2 where the random x is simply sent to all parties without considering any abort. In practice such protocols are impossible to obtain without a trusted third party, as a malicious adversary can usually first check the value of the random bit, and abort before letting the other party aware of this value. For this reason, we need to add an exchange of messages to allow the adversary to abort without letting the other parties know the value of the random bit.

We define in Protocol 4.1 the protocol $\{\tau_i\}_{i\in[n]}$ that realizes $\mathcal{V}_{|G\rangle}$ from $\mathcal{V}^f_{|G\rangle}$ and $\mathcal{R}_{\mathsf{CoinFlip}}$. We prove

Resource 4.2 Ideal resource $\mathcal{R}_{\mathsf{CoinFlip}}$

1. Sample a random bit string $x \xleftarrow{\$} \{0,1\}^n$
2. For each party $i \in [n]$, receive $c_i \in \{\top, \bot\}$. If any $c_i = \bot$, send x to party i, and wait for another abort bit $c'_i \in \{\top, \bot\}$: if $c'_i = \bot$, abort by sending $\bot$ on all interfaces, and otherwise send x to all parties.

We define the associated filter that, connected to interface i, sets $c_i = \top$ and forwards the x to the outer interface.

Protocol 4.1 $\{\tau_i\}_{i\in[n]}$ Realizing $\mathcal{V}_{|G\rangle}$ from $\mathcal{V}^f_{|G\rangle}$ and $\mathcal{R}_{\mathsf{CoinFlip}}$

1. For each honest party $i \in H$, τ_i receives from $\bot_H \mathcal{V}^f_{|G\rangle}$ a share of a quantum state $|G\rangle$ or an abort message (in which case they output $\bot$ and abort).
2. Each honest party $\{\tau_i\}_{i\in H}$ asks from $\mathcal{R}_{\mathsf{CoinFlip}}$ a message x or an abort bit (in which case they output $\bot$ and abort).
3. Each honest party $\{\tau_i\}_{i\in H}$ applies the stabilizer $\mathbf{X}^{x_i}\mathbf{Z}^{(Gx)_i}$ on their qubit, and outputs the resulting qubit.

now that Protocol 4.1 realizes $\mathcal{V}_{|G\rangle}$.

Theorem 4.1. *Protocol 4.1 realizes $\mathcal{V}_{|G\rangle}$.*

Proof sketch. The full proof is in appendix, but the main idea of the security proof is as follows: first, the simulator will send to the distinguisher the state sent by $\mathcal{V}_{|G\rangle}$ after partially applying on it a random stabilizer x. Then, the simulator will receive back a set of corrections from the distinguisher to apply on the side of the honest parties: since the simulator cannot apply these corrections as the qubits are on the side of the functionality, we use the fact that for stabilizer states, we can instead apply these corrections on the side of the adversary. Unfortunately, the simulator has also no access to the quantum register of the adversary: instead, the simulator adds this additional correction to x, and sends this to the distinguisher, pretending it was the random stabilizer sampled by $\mathcal{R}_{\mathsf{CoinFlip}}$. We provide an extensive analysis of this simulator in the full proof in Appendix B.2. □

Corollary 4.2. *Assuming the existence of a protocol for graph state verification fulfilling properties described in Theorem 3.2 and a coin flipping protocol realizing $\mathcal{R}_{\mathsf{CoinFlip}}$, there exists a protocol realizing $\mathcal{V}_{|G\rangle}$.*

See proof in Appendix B.2.

5 Use cases

Our result readily applies to some already existing protocols in the literature. In the following we show two such use cases and discuss their implication and improvement over previous works.

5.1 Generic translation tools

Before studying specific protocols, it is handy to first derive some generic theorems in order to translate between the notations used in existing graph state verification protocols and the notion of distance used in Definition 3.1. Indeed, while our notion of distance uses a single parameter ε quantifying the average distance to the ideal graph state (which is handy since in AC we have a single parameter ε' used to denote the distance between two systems $A \approx_{\varepsilon'} A$), existing works consider instead two parameters, namely the probability δ of being η-close to the ideal graph state.

Remark 5.1. *We also emphasize that most existing protocols implicitly assume that either all parties abort or that they all accept the final state together. We will refer to that property as* simultaneous abortion. *While Definition 3.1 does not strictly enforce this behavior, any protocol that does not fulfill simultaneous abortion with high probability will also have low guarantee in term of security. Said differently, the ε obtained in Definition 3.1 and Theorem 3.2 will be far from 0. This makes sense since our resource Resource 3.1 always sends the same abort/accept bit to all parties. So, if the adversary can send different abort bits to different parties, this gives directly a simple way to distinguish the ideal world from the real world, by simply checking if all parties share the same abort bit.*

This could be a real issue in practice, for instance in [PCW+11] if a malicious verifier is picked, the verifier could send different abort bits to all parties. As a result, without further checks, this protocol could only be proven composably secure for some constant *ε (as a reminder, in the original protocol, we expect to be able to get ε as small as wanted, the running time scaling polynomially with $O(1/\varepsilon)$). We have two options to avoid this issue:*

- *Either change the definition of $\mathcal{V}_{|G\rangle}$ to get rid of simultaneous abortion, meaning that some honest parties could abort while others would not.*
- *Or we could slightly adapt the protocol using a* broadcast channel, *i.e. a channel where all parties receive the same bit, in order to notify to all parties at the same time the final abort bit.*

Since the first option makes the resource harder to use, we will opt for the second solution. This makes even more sense as in existing protocols, the protocol already needs to obtain a random string known to all parties, which often already implicitly requires a broadcast channel. One might also ask whether, reciprocally, a broadcast channel is needed to realize $\mathcal{V}_{|G\rangle}$. It turns out that $\mathcal{V}^f_{|G\rangle}$ can already be (ab)used to obtain a kind of broadcast channel: since all parties see the same abort bit, one could decide that "abort" means 0 and that "accept" means 1. Therefore, it should come at no surprise that a broadcast channel is needed to realize $\mathcal{V}_{|G\rangle}$.

Lemma 5.2. *Let $\Pi = \{\pi_i\}_{i\in[n]\cup\mathsf{S}}$ be a protocol generating, when all parties are honest, a state $|G\rangle$ shared among all parties but the source. We assume that Π has simultaneous abortion (Remark 5.1), i.e. that for any adversary $\mathcal{A}$, either all honest parties abort at the same time with some probability $1-p$ or accept and output the averaged state $\rho_\top$. Then, if* any *of the following conditions is fulfilled, this protocol is an ε-graph state verification protocol according to Definition 3.1:*

- *If $p(1 - F(\rho_\top, \mathrm{Tr}_{[n]\setminus H}(|G\rangle\langle G|))) \leq \varepsilon$. This denotes the fact that the probability of accepting and outputting a state far from $|G\rangle$ to the honest parties is small.*
- *Or if $F(\rho_\top, \mathrm{Tr}_{[n]\setminus H}(|G\rangle\langle G|)) \geq 1 - \varepsilon$. This corresponds to the protocol's property to create a state close to the desired state. Note that this condition is strictly stronger than the first one, since an*

adversary might be able to produce such a state with negligible probability. Yet, unconditionally secure protocols might prefer this formulation.

See proof in Appendix B.3.

Lemma 5.3. *If a protocol has simultaneous abortion (Remark 5.1), and if the probability (on the randomness of $\mathcal{A}$ and the whole protocol) to have no abort and a final state far from the target $|G\rangle$ is small, more formally:*

$$\Pr\left[\operatorname{Tr}_{[n]\setminus H}|\psi_i\rangle \neq \left|\perp^H\right\rangle \wedge \sqrt{1-F^2(\operatorname{Tr}_{[n]\setminus H}|\psi_i\rangle\langle\psi_i|, \operatorname{Tr}_{[n]\setminus H}|G\rangle\langle G|)} \geq \eta \;\middle|\; |\psi_i\rangle \leftarrow \pi_H \mathcal{R}\mathcal{A}\right] \leq \delta \quad (18)$$

or, equivalently,

$$\Pr\left[\operatorname{Tr}_{[n]\setminus H}|\psi_i\rangle \neq \left|\perp^H\right\rangle \wedge \min_U \mathrm{TD}((I^H \otimes U^M)|\psi_i\rangle, |G\rangle) \geq \eta \;\middle|\; |\psi_i\rangle \leftarrow \pi_H \mathcal{R}\mathcal{A}\right] \leq \delta \quad (19)$$

then this protocol is an $(\delta+\eta^2)$-graph state verification protocol according to Definition 3.1.

See proof in Appendix B.3.

5.2 GHZ-state verification

In [PCW+11], the authors develop and analyze an n-party verification protocol consisting only of classical communication and local quantum operations once the state is shared. One of the parties, called the *Verifier*, has a central role in the protocol: it sends instructions to all parties and broadcasts the output of the verification. We recall the protocol of [PCW+11] in Protocol 5.1. In the AC language, it uses quantum and classical authenticated secret channels, a broadcast channel and a Common Random String resource (CRS) as concrete resources to build an ideal GHZ-state sharing functionality. Note that it is important that the random string is sent by the CRS once all parties received the state from the source.

This protocol has been extensively studied and presents desirable properties: notably the probability of aborting increases when the state sent by the source is further away from the target state (locally equivalent to) $|GHZ\rangle$, and this probability can be made arbitrary small by increasing the number of rounds. More precisely, for one round, if ρ denotes the state shared among the parties by the source, and if the verifier is honest, the probability of aborting is $\Pr[\, b_{out} = 1\,] = \frac{\tau^2}{4}$ with:

$$\tau = \min_U \mathrm{TD}(|GHZ\rangle\langle GHZ|, U\rho U^\dagger) \quad (20)$$

where TD is the trace distance and U is an operator acting only on the space of the dishonest parties. Since the verifier might not always be honest, we can repeat this protocol to increase the abort probability when the state is maliciously prepared. The verification protocol thus consists in rounds where the source shares a state to the parties, who then samples $r \xleftarrow{\$} \{0,1\}^S$ to decide if they verify the state or if they keep it. This process goes on until until either $r = 0\ldots0$ in step (3), in which case the parties output a quantum state, or if $b_{out} = 1$ in step (8), in which case the parties abort. The fact that the parties decide to verify or keep the state *after* it is shared ensures that possible malicious parties do not adapt their behaviour to a particular round. In [PCW+11], the authors prove that the probability that a state $|\psi\rangle$ is accepted after repeating the protocol until $r = 0\cdots0$, and $|\psi\rangle$ is further than ε from the GHZ state is given by:

$$\Pr\left[|\psi\rangle \text{ accepted} \wedge \min_U \mathrm{TD}(U|\Psi\rangle, |GHZ\rangle) \geq \epsilon\right] = 2^{-S}\frac{4n}{h\varepsilon^2} \quad (21)$$

where U acts on the space of the dishonest parties, n is the total number of parties and h is the number of honest parties.

In later work [MPB+16], an experimental realization of a loss-tolerant variant of this protocol has been implemented, hinting towards practicability of this protocol in real-life networks. Note that the

Protocol 5.1 Multipartite entanglement verification protocol

1. The source creates an n-qubit state locally equivalent to the GHZ state (up to local Hadamard and phase shift $\sqrt{Z}$ gate) and sends each qubit i to party i.

2. After receiving the state, the parties receive $r \xleftarrow{\$} \{0,1\}^S$ and $i \xleftarrow{\$} [n]$ from a CRS resource, where S is a security parameter. Note that the source should not reveal r and i before the state is received by all parties.

3. If $r = 0 \dots 0$, the state received at step 1 is outputted by each party and the protocol stops.

4. Otherwise, if $r \neq 0 \dots 0$, party i is designed as the Verifier.

5. The Verifier selects for each $i \in [n]$ a random input $x_i \in \{0,1\}$ such that $\sum_{i=1}^{n} x_i \equiv 0 \mod 2$ and sends it to the corresponding party via an authenticated private classical channel resource. The Verifier keeps one to themselves.

6. If $x_i = 0$, party i performs a Z operation on their qubit. If $x_i = 1$, party i performs a Hadamard operation.

7. Each party i measures their qubit in the $\{|0\rangle, |1\rangle\}$ basis and sends their outcome y_i to the Verifier via the classical channel.

8. The Verifier accepts and broadcasts $b_{out} \coloneqq 0$ if and only if

$$\sum_{i=1}^{n} y_i \equiv \frac{1}{2} \sum_{i=1}^{n} x_i \mod 2$$

   Otherwise, the Verifier broadcasts $b_{out} = 1$ and the protocol aborts.

9. If the Verifier has not rejected, we restart from Step 1.

result that we show in this section also applies to this variant.

A previous composable security study of this protocol [YDK21] could not show the composable security of this protocol in the case of a coalition on malicious party and source and conjectured that it was impossible to prove in the AC framework. Our present work answers this conjecture by the negative. The key difference with the previous work lies in Remark 2.2, which allows local simulators to communicate between each others via an additional communication channel resource. Equivalently, this allows to consider a single global simulator having access to all the corrupted interfaces of the ideal result, which is actually the choice made by the Universal Composability framework. Building on the work from the previous sections, we can show the following lemma proving composable security of the protocol from [PCW+11]:

Lemma 5.4. *The protocol defined in [PCW+11], assuming that we use a broadcast channel to transmit the abort bit, is an ε-graph state verification protocol for $\varepsilon := \frac{1}{2^{S/2}}(4n+1)$, where 2^S is the average number of tests before outputting a state as defined in [PCW+11] and n is the total number of parties. As a result, it $(2\sqrt{2\varepsilon-\varepsilon^2})$-realizes $\mathcal{V}^f_{|G\rangle}$ as defined in Theorem 3.2, and can be turned into a protocol that $(2\sqrt{2\varepsilon-\varepsilon^2})$-realizes $\mathcal{V}_{|G\rangle}$.*

See proof in Appendix B.3.

5.3 Graph state verification

The work of [UM22] presents a generalization of the previous protocol to arbitrary graph states, where the verifier will ask parties to measure random stabilizers of the graph state. We can similarly show that it is composably secure. Note however that this protocol, like the previous one, shows security scaling polynomially with the security parameter while we usually expect it to scale super-polynomially. However, this allows significantly simpler protocols and most existing graph state verification protocols have this property. But our framework, of course, also applies to protocols that are super-polynomially secure.

Lemma 5.5. *We define, as in the theorem 3 of [UM22] (where we use the fact that conditioned on non-aborting, we have $N_{\mathsf{pass}} \geq \lambda J N_{\mathsf{test}} - \frac{N_{\mathsf{test}}}{2J}$ as described in protocol 2 to simplify the expression of p_0 and avoid any dependency on a number that might be different every time we run the protocol):*

- *$J = 2^n$ or $J = n$ depending on G as described in [UM22, Thm. 2],*
- *λ be the security parameter growing polynomially with the number of tests*
- *m and c some positive constants chosen so that p_0 and η_0 defined later are greater than 0,*
- *$p_0 := [1 - \sum_{x=0}^{\lambda}(1-\frac{1}{n})^x(\frac{1}{n}J^{\frac{-2cm}{3}})^{\lambda-x}]^J$ (we got rid of the number of honest parties $|H| \geq 1$ since we want this to be independent of the number of malicious parties)*
- *$\eta_0 := \left(\frac{1}{\lambda} - \frac{1}{\lambda^2}\right) + \left(1+\frac{1}{\lambda}\right)\frac{\sqrt{c}+1/2}{J}$*

The symmetric protocol 2 defined in [UM22], assuming that we use a broadcast channel to transmit the abort bit, is an ε-graph state verification protocol for $\varepsilon := 1 - p_0 + 2\eta_0 - \eta_0^2$. As a result, it $(2\sqrt{2\varepsilon-\varepsilon^2})$-realizes $\mathcal{V}^f_{|G\rangle}$ as defined in Theorem 3.2, and can be turned into a protocol that $(2\sqrt{2\varepsilon-\varepsilon^2})$-realizes $\mathcal{V}_{|G\rangle}$.

See proof in Appendix B.3.

6 Conclusion

In this work, we studied the composable security of generic graph-state verification protocols, i.e. protocols consisting of a source sharing graph-states to a network of parties, who then only perform local operations

and classical communications, to decide whether they can use some of the states for another protocol. We showed that they can be considered equivalent to an ideal abstract resource, up to an ε that we explicited. This resource shares graph-states while allowing a restricted class of corrections to be applied to them by malicious parties. We proved that without modifying the original protocols, these malicious actions cannot be prevented when considering generic black-box simulators. We thus showed how to modify the original verification protocols to prevent the malicious parties to effectively act on the shared graph-state. The modification consists in asking the parties to jointly apply a random stabilizer to the state that they receive. By doing so, the modified protocol can be considered equivalent to a ideal functionality simply sharing graph-states. In this modified version, the dishonest parties can only force the protocol to abort.

To prove our results, we studied the class of *mergeable states*, to which graph-states belongs. We showed that we can merge two copies of a graph state into one copy, by acting only on a partition of the qubits. This generalizes entanglement swapping to arbitrary graph state. We explicited the measurements and local operations to do this swapping using scalable ZX-calculus [CHP19]. For graph-state manipulation, the scalable ZX-calculus formalism proved to be more handy than the usual density matrix formalism, and we provide an introduction in Appendix A. We believe this work is one of the first using such methods in the context of quantum cryptography.

We emphasize that our results mostly preserves the security features of the original protocol. Notably, if the initial protocol is ε-secure, our claim is that it ε'-realizes the above functionalities where ε' is polynomially related to ε. In particular, if ε is scaling inverse polynomially with the security parameter, ε' will also scale inverse polynomially. On the other hand if ε is negligible, so will ε'.

Our results are crucial in the context of network protocol development. The class of verification protocols that we studied are use as building-blocks in many quantum network protocols. Our work explicits to which extent they can be repeatedly used by computing of communication protocols to get verified graph-state without threatening the overall security. We showed in the last section of the paper how to apply our result to two existing verification protocols [PCW+11, UM22]. In the process, we answered negatively a conjecture posed in [YDK21] stating that graph-state verification protocols cannot be proven composably secure in the Abstract cryptography framework. Moreover, our composability proof readily applies in a network where trusted parties wish to build trust on a source of graph-state. This is the case in many real-life scenario where a group of people wish to get a graph-state from an untrusted network to perform, for example, a multiparty computing protocol. In future works, we will study different contexts in which our result can apply.

7 Acknowledgment

The authors deeply thank Robert Booth for suggesting a more elegant representation of graph states. This work is co-funded by the European Union (ERC, ASC-Q, 101040624) and the Horizon Europe program (QUCATS). It is also supported by the Dutch National Growth Fund (NGF), as part of the Quantum Delta NL programme, and the Government of Spain (Severo Ochoa CEX2019-000910-S and FUNQIP), Fundació Cellex, Fundació Mir-Puig, Generalitat de Catalunya (CERCA program). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council. This work is supported by France 2030 under the French National Research Agency projects HQI ANR-22-PNCQ-0002 and EPiQ ANR-22-PETQ-0007 as well as ANR Project SecNISQ. Neither the European Union nor the granting authority can be held responsible for them.

References

[Aar09] S. Aaronson. Quantum Copy-Protection and Quantum Money. In *2009 24th Annual IEEE Conference on Computational Complexity*. 2009 24th Annual IEEE Conference on Computational Complexity, pages 229–242, July 2009.

[BBC+93] C. H. Bennett, G. Brassard, C. Crépeau, R. Jozsa, A. Peres, and W. K. Wootters. Teleporting an unknown quantum state via dual classical and Einstein-Podolsky-Rosen channels. *Physical Review Letters*, 70(13):1895–1899, March 29, 1993.

[BBD+09] H. J. Briegel, D. E. Browne, W. Dür, R. Raussendorf, and M. Van den Nest. Measurement-based quantum computation. *Nature Physics*, 5(1):19–26, January 2009.

[BCK+20] M. Bozzio, U. Chabaud, I. Kerenidis, and E. Diamanti. Quantum weak coin flipping with a single photon, 2020.

[BFK09] A. Broadbent, J. Fitzsimons, and E. Kashefi. Universal Blind Quantum Computation. In *2009 50th Annual IEEE Symposium on Foundations of Computer Science*. 2009 50th Annual IEEE Symposium on Foundations of Computer Science, pages 517–526, October 2009.

[BOT+18] M. Bozzio, A. Orieux, L. Trigo Vidarte, I. Zaquine, I. Kerenidis, and E. Diamanti. Experimental investigation of practical unforgeable quantum money. *npj Quantum Information*, 4(1), January 2018.

[Can01] R. Canetti. Universally composable security: a new paradigm for cryptographic protocols. In *Proceedings 42nd IEEE Symposium on Foundations of Computer Science*. Proceedings 42nd IEEE Symposium on Foundations of Computer Science, pages 136–145, October 2001.

[Car20] T. Carette. A note on diagonal gates in SZX-calculus, December 17, 2020.

[CD08] B. Coecke and R. Duncan. Interacting Quantum Observables. In L. Aceto, I. Damgård, L. A. Goldberg, M. M. Halldórsson, A. Ingólfsdóttir, and I. Walukiewicz, editors, *Automata, Languages and Programming*, Lecture Notes in Computer Science, pages 298–310, Berlin, Heidelberg. Springer, 2008.

[CDK22] F. Centrone, E. Diamanti, and I. Kerenidis. Practical quantum electronic voting. *Phys. Rev. Applied*, 18:014005, 2022.

[CHP19] T. Carette, D. Horsman, and S. Perdrix. SZX-calculus: Scalable Graphical Quantum Reasoning. 2019.

[CK17] B. Coecke and A. Kissinger. *Picturing Quantum Processes: A First Course in Quantum Theory and Diagrammatic Reasoning*. Cambridge University Press, Cambridge, 2017.

[CMS23] L. Colisson, G. Muguruza, and F. Speelman. Oblivious Transfer from Zero-Knowledge Proofs, or How to Achieve Round-Optimal Quantum Oblivious Transfer and Zero-Knowledge Proofs on Quantum States, March 2, 2023. To appear in ASIACRYPT 2023.

[CW05] M. Christandl and S. Wehner. Quantum Anonymous Transmissions. In B. Roy, editor, *Advances in Cryptology - ASIACRYPT 2005*, Lecture Notes in Computer Science, pages 217–235, Berlin, Heidelberg. Springer, 2005.

[DGJ+20] Y. Dulek, A. B. Grilo, S. Jeffery, C. Majenz, and C. Schaffner. Secure Multi-party Quantum Computation with a Dishonest Majority. In A. Canteaut and Y. Ishai, editors, *Advances in Cryptology – EUROCRYPT 2020*, Lecture Notes in Computer Science, pages 729–758, Cham. Springer International Publishing, 2020.

[DNS12] F. Dupuis, J. B. Nielsen, and L. Salvail. Actively Secure Two-Party Evaluation of Any Quantum Operation. In R. Safavi-Naini and R. Canetti, editors, *Advances in Cryptology – CRYPTO 2012*, Lecture Notes in Computer Science, pages 794–811, Berlin, Heidelberg. Springer, 2012.

[FK17] J. F. Fitzsimons and E. Kashefi. Unconditionally verifiable blind quantum computation. *Physical Review A*, 96(1):012303, July 5, 2017.

[Gan09] M. Ganz. Quantum leader election. *Quantum Information Processing*, 16, October 2009.

[GKK19] A. Gheorghiu, T. Kapourniotis, and E. Kashefi. Verification of Quantum Computation: An Overview of Existing Approaches. *Theory of Computing Systems*, 63(4):715–808, May 1, 2019.

[GLS+21] A. B. Grilo, H. Lin, F. Song, and V. Vaikuntanathan. Oblivious Transfer Is in MiniQCrypt. In A. Canteaut and F.-X. Standaert, editors, *Advances in Cryptology – EUROCRYPT 2021*, Lecture Notes in Computer Science, pages 531–561, Cham. Springer International Publishing, 2021.

[HZB+06] M. Hillery, M. Ziman, V. Bužek, and M. Bieliková. Towards quantum-based privacy and voting. *Physics Letters A*, 349(1):75–81, January 9, 2006.

[KP17] E. Kashefi and A. Pappa. Multiparty Delegated Quantum Computing. *Cryptography*, 1(2):12, July 2017.

[Mau11] U. Maurer. Constructive cryptography - a new paradigm for security definitions and proofs. *IN Theory of Security and Applications*:33–56, 2011.

[Mau12] U. Maurer. Constructive Cryptography – A New Paradigm for Security Definitions and Proofs. In S. Mödersheim and C. Palamidessi, editors, *Theory of Security and Applications*, Lecture Notes in Computer Science, pages 33–56, Berlin, Heidelberg. Springer, 2012.

[MF18] T. Morimae and J. F. Fitzsimons. Post hoc verification with a single prover. *Physical Review Letters*, 120(4):040501, January 22, 2018.

[MPB+16] W. McCutcheon, A. Pappa, B. Bell, A. McMillan, A. Chailloux, T. Lawson, M. Mafu, D. Markham, E. Diamanti, I. Kerenidis, J. Rarity, and M. Tame. Experimental verification of multipartite entanglement in quantum networks. *Nature Communications*, 7:13251, November 2016.

[MR11a] U. Maurer and R. Renner. Abstract Cryptography. In *ICS*, 2011.

[MR11b] U. Maurer and R. Renner. Abstract cryptography. *In Innovations In Computer Science*, 2011.

[MR16] U. Maurer and R. Renner. From indifferentiability to constructive cryptography (and back). In *Theory of Cryptography*, pages 3–24, Berlin, Heidelberg. Springer Berlin Heidelberg, 2016.

[NC10] M. A. Nielsen and I. L. Chuang. Quantum Computation and Quantum Information: 10th Anniversary Edition. Higher Education from Cambridge University Press. December 9, 2010.

[PCW+11] A. Pappa, A. Chailloux, S. Wehner, E. Diamanti, and I. Kerenidis. Multipartite entanglement verification resistant against dishonest parties. *Physical Review Letters*, 108, December 2011.

[UM22] A. Unnikrishnan and D. Markham. Verification of graph states in an untrusted network. *Physical Review A*, 105(5):052420, May 13, 2022.

[UMY+18] A. Unnikrishnan, I. MacFarlane, R. Yi, E. Diamanti, D. Markham, and I. Kerenidis. Anonymity for practical quantum networks. *Physical Review Letters*, 122, November 2018.

[Unr10] D. Unruh. Universally Composable Quantum Multi-party Computation. In H. Gilbert, editor, *Advances in Cryptology – EUROCRYPT 2010*, Lecture Notes in Computer Science, pages 486–505, Berlin, Heidelberg. Springer, 2010.

[van20] J. van de Wetering. ZX-calculus for the working quantum computer scientist. *arXiv:2012.13966 [quant-ph]*, December 2020.

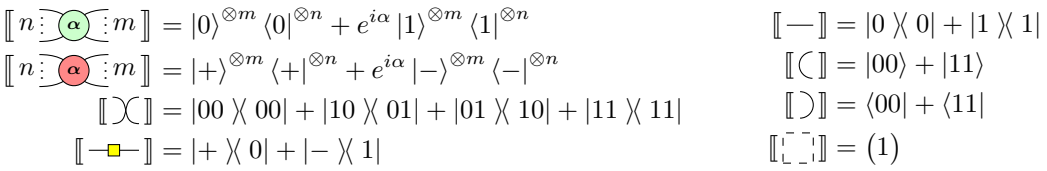
[Wie83] S. Wiesner. Conjugate coding. *ACM SIGACT News*, 15(1):78–88, January 1, 1983.

[YDK21] R. Yehia, E. Diamanti, and I. Kerenidis. Composable security for multipartite entanglement verification. *Physical Review A*, 103(5):052609, May 19, 2021.

A Introduction to the (scalable) ZX-calculus

A.1 Introduction to the (scalable) ZX-calculus.

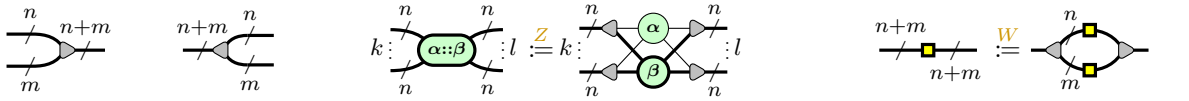
ZX-diagrams. A ZX-diagram is a graph with some inputs (on the left) and outputs (on the right). We can associate to any ZX-diagram a matrix, using the following interpretation of the generators, where the colored nodes are being called "spiders", and H being strictly speaking only a syntactic sugar:



Note that when $\alpha = 0$, we can omit the angle in the spiders. Moreover, we can compose these generators sequentially (resp. in parallel). We then obtain the resulting natural interpretation, by inductively computing the matrix product (resp. tensor product) of the interpretation of the sub-diagrams.

Circuit to ZX. One can turn any quantum circuit into a ZX-diagram. Let $(a, b) \in \{0, 1\}^2$ and $\alpha \in \mathbb{R}$. Then, up to a non-relevant re-normalisation scalar and global phase, we represent basic states using $-\overset{a\pi}{\bullet} = |a\rangle$, $-\overset{a\pi}{\circ} = H|a\rangle$ and one-qubits gates using $-\square- = \mathbf{H}$, $-\overset{\alpha}{\bullet}- = \mathbf{R}_x(\alpha)$ and $-\overset{\alpha}{\circ}- = \mathbf{R}_z(\alpha)$. We have in particular $-\overset{a\pi}{\bullet}- = \mathbf{X}^a$ and $-\overset{a\pi}{\circ}- = \mathbf{Z}^a$. Two qubit gates are represented as[8] $\overset{\bullet}{\underset{\circ}{|}} = \mathbf{CNOT}$ and $\overset{\circ}{\underset{\circ}{|\square|}} = \wedge\mathbf{Z}$. Moreover, we represent a measurement in the computational basis whose outcome is a using $-\overset{a\pi}{\bullet} = \langle a|$ and a measurement in the Hadamard basis, i.e. the projection on $|0\rangle + (-1)^a|1\rangle$, whose outcome is a $-\overset{a\pi}{\circ} = \langle a|H$. Finally, $\overset{a\pi}{\underset{b\pi}{\frown}}$ represents a Bell measurement (projection on $|0b\rangle + (-1)^a|1\bar{b}\rangle$) with outcomes (a, b).

Scalable ZX. The scalable ZX-calculus [CHP19] generalises these generators: wires can be grouped (or ungrouped) together. We represent these grouped wires as bold wires, where we specify above the number of wires in the group when there can be a confusion. The grouping/ungrouping of wires is done using so-called gatherers and dividers, shown in the first two diagrams of Eq. (22) below. In scalable ZX, spiders can contain lists of real numbers, corresponding to stacked spiders, as we show for example in the two last diagram of Eq. (22) with $\alpha \in \mathbb{R}$ and $\beta \in \mathbb{R}^{n-1}$:



(22)

The interest of the ZX-calculus is that we can rewrite a ZX diagram using some rewriting rules while preserving the interpretation of the corresponding matrix. The first rule is that *only connectivity matters*, i.e. we can bend wires arbitrarily as soon as the corresponding undirected graph is left unchanged. Other rules and theorems are described in Fig. 9. Note that, for simplicity, we will remove all scalars, i.e. sub-graphs having no input nor output, as they correspond to global phases (not observable physically) and/or a re-normalisation of the state.

Lemma A.1 (Rewiring rule R [CHP19, Thm. 3.2])**.** *Two diagrams composed only of identity, gatherers and dividers are equal iff their respective number of inputs and outputs are equal.*

[8] Although the generators do not allow vertical wires, we will see that we can freely bend wires and move nodes since "only topology matters". We can therefore consider ZX-diagrams as undirected graphs.

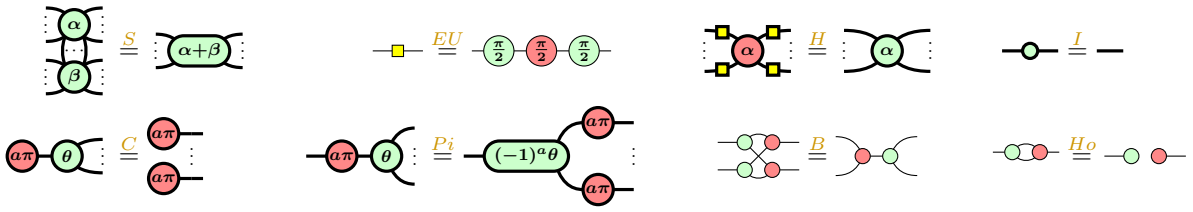
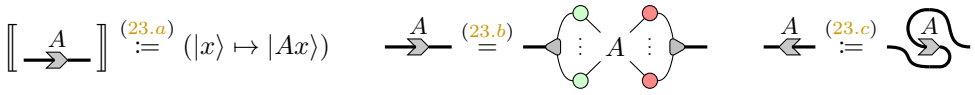


Figure 9: Rules and theorems of the SZX calculus (the rules also hold when read from right to left and after exchanging colors, note that we omit the more generic Euler rule as it is not needed for the Clifford fragment that we use here). $a \in \{0,1\}^n$ and $b \in \{0,1\}^n$ are arbitrary vectors of binary variables, $\theta \in \mathbb{R}^n$ is an arbitrary vector, and operations between vectors are component-by-component. In the S rule, at least one connection must exist between both spiders, and bold wires indicate wires of size ≥ 1.

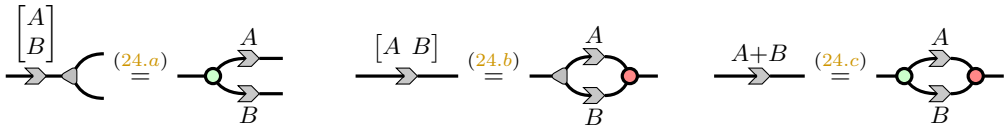
The scalable ZX-calculus also provides a way to use arbitrary binary matrices to specify the connectivity between nodes: for any matrix $A \in \mathbb{Z}_2^{m \times n}$, we have:



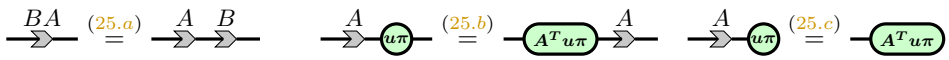
(23)

Where, in the second equation, A represents the biadjency matrix of the bipartite green/red graph.

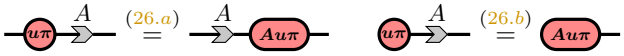
For any binary vector u and binary matrix A and B of appropriate size, we also have the following rewriting rules, that informally come from the fact that a green spider copies states in the computational basis while a red spider performs the modulo sum of its inputs:



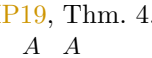
(24)

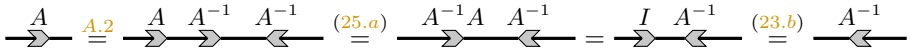


(25)



(26)

Lemma A.2 ([CHP19, Thm. 4.8, 4.9]). *If A is injective, we have*  *and similarly, if A is surjective, we have*  . *In particular, if A is bijective, we have:*

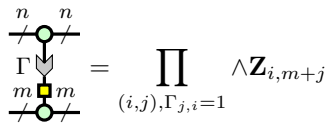


(27)

A.2 Graph states

We list below some basic properties to graphically represent and manipulate graph states. Most of these facts and proofs can be found in [Car20, CHP19].

Lemma A.3. *Let $\Gamma \in \mathbb{Z}_2^{m \times n}$, then the following ZX diagram applies a $\wedge\mathbf{Z}$ between the i-th qubit of the first group of qubits and the j-th qubit of the second group of qubits iff $\Gamma_{j,i} = 1$, or, more formally:*



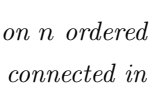
$$= \prod_{(i,j),\Gamma_{j,i}=1} \wedge\mathbf{Z}_{i,m+j} \tag{28}$$

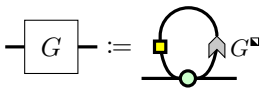
where $\wedge\mathbf{Z}_{i,j}$ applies a $\wedge\mathbf{Z}$ between the i-th and j-th qubit, assuming i, j and the qubits are all indexed starting from 0.

Definition A.4. *Let $G = (V, E)$ be an undirected graph on n ordered vertices in V. By slightly abusing notations, we will also denote its representation in term of an adjacency matrix as $G \in \mathbb{Z}_2^{n\times n}$, the rows and columns of G being indexed by elements in V, where $G_{i,j} = 1$ iff there exists an edge between i and j and $G_{i,j} = 0$ otherwise. We also denote $G^{◹}$ and $G^{◺}$ as, respectively, the upper and lower triangular matrix of G (note that G has zeros on its diagonal). In particular, $G = G^{◹} + G^{◺}$, and since $G^T = G$, we also have $(G^{◹})^T = G^{◺}$. For any partition (A, B) of V of G (for simplicity, we assume that elements in V are ordered so that elements in A appear before elements in B), we define $G_A \in \mathbb{Z}_2^{|A|\times|A|}$, $G_B \in \mathbb{Z}_2^{|B|\times|B|}$, and $\Gamma_{A\to B} \in \mathbb{Z}_2^{|B|\times|A|}$ (or simply Γ) such that:*

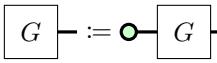
$$G = \begin{bmatrix} G_A & \Gamma^T \\ \Gamma & G_B \end{bmatrix} \tag{29}$$

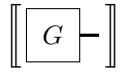
In particular, G_A and G_B are the subgraphs of G restricted to vertices in A and B respectively, and Γ is the biadjency matrix between elements in G_A and elements in G_B.

Definition A.5. *For any undirected graph $G = (V, E)$ on n ordered vertices, we define*  *as the operation that applies a $\wedge\mathbf{Z}$ gate between all input qubits connected in G (the wires being ordered following the order on V). We can represent this operation diagrammatically (see [Car20]) as:*

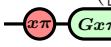


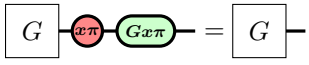
(30)

Moreover, we define the graph states G as  .

Lemma A.6. *For any graph G,*  $= |G\rangle$.

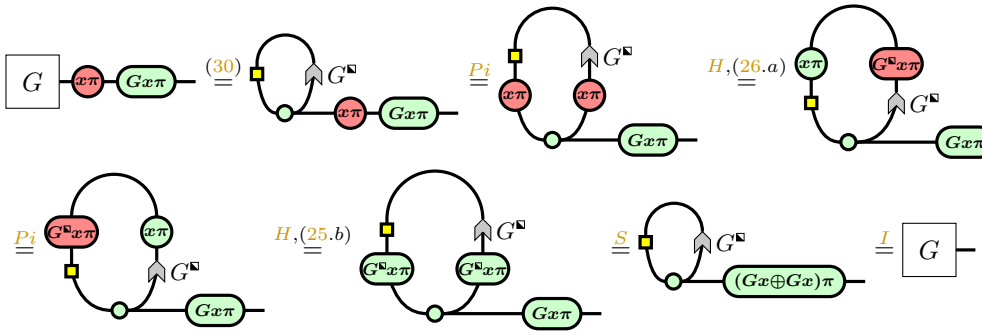
The proof of this statement can be found in [Car20], but the main idea is to show that this state is stabilized by the stabilizers of $|G\rangle$, which is formalized by the following lemma:

Lemma A.7 ([Car20])**.** *Let $G = (V, E)$ be a graph, and $x \in \mathbb{Z}_2^{|V|}$ be a vector (indexed by vertices in V). Then,*  *is a stabilizer of G:*



(31)

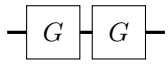
Proof. The proof of this statement can be found in [Car20], but we rewrite it here for completeness:



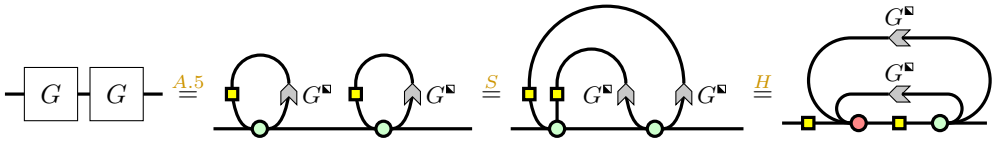
(32)

(33)

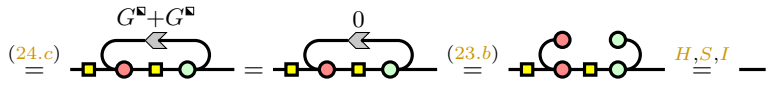
□

Lemma A.8. *For any graph G,*  .

Proof. This fact is a direct consequence of the property $\wedge\mathbf{Z}\wedge\mathbf{Z} = \mathbf{I}$. More formally:

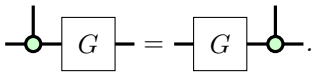


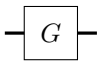
(34)



(35)

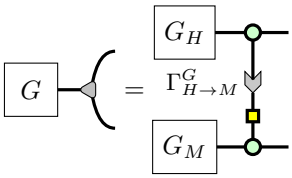
□

Lemma A.9. *For any graph G,*  .

Proof. This is a direct application of the spider rule on the definition of  . □

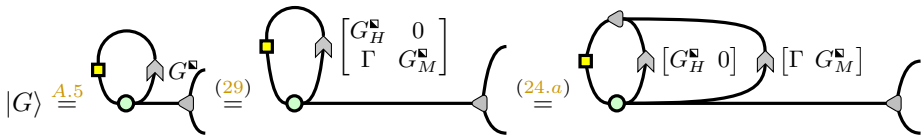
We describe now a way to partition a graph G into two subgraphs.

Lemma A.10. *Let $G = (V, E)$. Let $H \subseteq V$ and $M \coloneqq V \setminus H$ be a partition of V (for simplicity we assume that we re-order elements in V so that elements in H are smaller than elements in M). Then, using notations from Definition A.4, we have:*

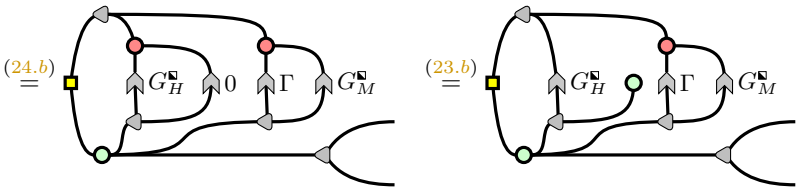


(36)

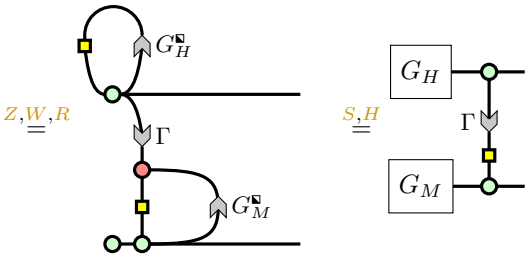
Proof. Intuitively, this lemma only means that in order to create G, we can first create the graph states G_H and G_M, and apply after $\wedge\mathbf{Z}$ gates between elements in G_H and G_M. This can be formalized diagrammatically using the decomposition rules for block matrices:



(37)



(38)



(39)

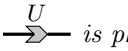
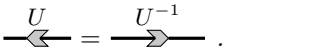
□

Lemma A.11. *Let $\Gamma \in \mathbb{Z}_2^{m\times n}$ be a binary matrix. Then, there exists an invertible matrix $U \in \mathbb{Z}_2^{n\times n}$, an integer r (the rank), a matrix $R \in \mathbb{Z}_2^{r\times n-r}$ and an invertible matrix $V \in \mathbb{Z}_2^{m\times m}$ such that:*

$$\Gamma = V \begin{bmatrix} I_r & R \\ \mathbf{0} & \mathbf{0} \end{bmatrix} U \tag{40}$$

where $I_r \in \mathbb{Z}_2^{r\times r}$ is the identity matrix and $\mathbf{0}$ is the zero matrix.

Proof. This is a direct consequence of the Gaussian elimination algorithm: using Gaussian elimination we can obtain a row echelon form (using only elementary row operations, i.e. swapping rows and adding a multiple of one row to another row). Then using column swap, we can reorder the row echelon form matrix to bring the pivots on the diagonal of the matrix. Finally, by subtracting for each row j (starting from the last row) the rows i (for any $i > j$) if j's row contains a one on i-th column, we can remove all elements above the diagonal, leading to the identity in the upper right corner. By combining all (invertible) elementary row and column operations into U and V, we obtain our final form. □

Lemma A.12. *Let $U \in \mathbb{Z}_2^{n\times n}$ be an invertible matrix, then*  *is physically implementable without auxiliary qubits using only* **CNOT** *and swap operations, and*  .

Proof. The fact that  is a direct consequence of [CHP19, Lem. 4.8, 4.9]. To see that it can be written as a sequence of **CNOT** and swap operations, we can first realize that since U is invertible, using the Gaussian elimination, we can find elementary row and column operations $E_1, \dots, E_n$ and $E'_1, \dots, \times E'_n$ such that:

$$E_1 \dots E_n U E'_1 \dots E'_n = I \tag{41}$$

The elementary row (resp. column) operations can either be:

- Operation 1: Multiply line (resp. column) j by a non-null scalar: since the only non-null scalar modulo 2 is 1, all operations of this form would be identity so we can remove such operations.
- Operation 2: Exchange the j-th line (resp. column) and the j'-th line (resp. column).
- Operation 3: Add the column j to column j' (in theory, we can multiply first the j-th line by an arbitrary factor, but since we are working modulo 2, the only interesting case is when this scalar is 1 since when it is equal to 0 nothing happens)

We can see that all these operations are self-inverse, so we have:

$$U = E_n \dots E_1 E'_n \dots E'_1 \tag{42}$$

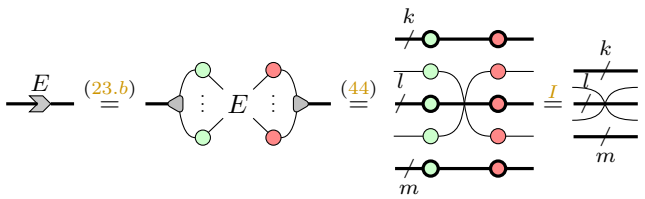
i.e.

 (43)

Moreover, any such operation can be implemented using a swap or a **CNOT** gate. If the elementary operation E is Operation 2, i.e. a swap, then E can literally be implemented by the same swap operation. This can be seen for instance by realizing that a swap can be realized via a matrix of this form:

$$E = \begin{bmatrix} \mathbf{1^k} & & & & \\ & 0 & & 1 & \\ & & \mathbf{1^l} & & \\ & 1 & & 0 & \\ & & & & \mathbf{1^m} \end{bmatrix} \tag{44}$$

where 1^k is the diagonal matrix of size $k \times k$ with ones on its diagonal, and $k = j - 1$, $l = j' - j - 1$. Therefore, using the characterization in Eq. (23), we have:

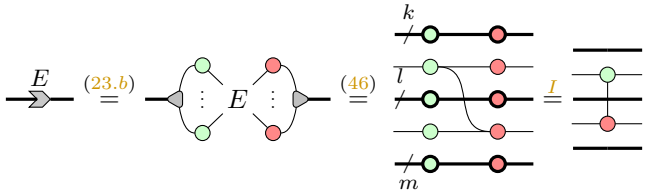


(45)

Similarly, if E is Operation 3, then we can write E as a matrix of this form (the 1 might be on the other side of the diagonal if $j' < j$):

$$E = \begin{bmatrix} \mathbf{1^k} & & & & \\ & 1 & & & \\ & & \mathbf{1^l} & & \\ & 1 & & 1 & \\ & & & & \mathbf{1^m} \end{bmatrix} \tag{46}$$

where $k = j - 1$, $l = j' - j - 1$ which gives, using the characterization in Eq. (23):



(47)

which corresponds exactly to a **CNOT** gate where the j-th qubit is the source and the j'-th qubit is the target. □

B Proofs

We details in this appendix some proofs of the main paper.

B.1 Proofs of Section 3

Theorem 3.3 (Impossibility of black-box realization of $\mathcal{V}_{|G\rangle}$)**.** *There exists some*[9] *graph states* $|G\rangle$ *such that for any graph state verification protocol* $\Pi \coloneqq \{\pi_i\}_{i\in[n]\cup\mathsf{S}}$ *using a resource* $\mathcal{R}$ *and producing* $|G\rangle$ *(i.e.* $\{\pi_i\}_{i\in[n]\cup\mathsf{S}}\mathcal{R}$ *outputs* $|G\rangle$ *shared among the* n *parties), it is* impossible *to prove that* Π *is* ε*-realizing* $\mathcal{V}_{|G\rangle}$ *(Resource 3.1) for any* $\varepsilon < 1/2$ *if the simulator is black-box*[10]*, in the sense that the simulator interacts with the interface controlled by the environment by running* $\{\pi_i\}_{i\in H}\mathcal{R}$*, forwarding all messages between the malicious interfaces of* $\mathcal{R}$ *and the environment.*

Proof of Theorem 3.3. This proof is diagrammatically illustrated with equations starting from Eq. (10). In the following, we will consider the graph state $|G\rangle \coloneqq |00\rangle + |11\rangle$, consisting of a Bell pair. By contradiction,

[9] Actually most graph states have this property as soon as they are not separable.

[10] We call it black-box since the definition of the simulator is mostly independent of the protocol, as it can only execute Π without having access to its code. This definition of black-box simulator is relatively generic: if the simulator does not know the definition of the protocol Π, it can basically only forward the messages of the protocol to the distinguisher. For instance, if all exchanged messages are signed with a key unknown to the simulator (but which is part of the public description of the protocol), the only way to communicate with the distinguisher is to forward the messages sent by running $\{\pi_i\}_{i\in H}\mathcal{R}$. We could formalize this by giving an even more generic definition of black-box simulator that is not explicitly asked to forward the messages of $\{\pi_i\}_{i\in H}\mathcal{R}$ to the distinguisher, but this would obfuscate our proof without clear benefits.

we assume the existence of a protocol $\Pi \coloneqq \{\pi_0, \pi_1, \pi_{\mathtt{S}}\}$ ε-realizing $\mathcal{V}_{|G\rangle}$, where the simulator σ is black-box as defined in Theorem 3.3. In particular, we can consider the case where parties 0 and $\mathtt{S}$ are honest and party 1 is corrupted. We also consider the following distinguisher $\mathcal{D}$:

- $\mathcal{D}$ runs the honest protocol π_1, interacting with the corrupted interfaces. At the end of the protocol, it gets a bipartite quantum state $\rho^{0,1}$ from the output from π_1 and from the honest interface of the functionality $\mathcal{V}_{|G\rangle}$.
- Then, $\mathcal{D}$ will measure both qubits in the computational basis, outputting 0 if both outcomes are equal and different from $\perp$ and 1 otherwise.

We will show that this distinguisher can distinguish the real world from the ideal world with an advantage greater than ε, raising a contradiction since both worlds must be indistinguishable against any distinguisher and any subset of corrupted parties.

First, if he distinguisher is interacting with the real world $\pi_{\{0,\mathtt{S}\}}\mathcal{R}$, because $\mathcal{D}$ is following the honest protocol, we have $\rho = \{\pi_i\}_{i\in[n]\cup\mathtt{S}}\mathcal{R}$. Since Π is correct by assumption (recall that $\{\pi_i\}_{i\in[n]\cup\mathtt{S}}\mathcal{R}$ outputs $|G\rangle = |00\rangle + |11\rangle$), we always have $\rho = |00\rangle + |11\rangle$, and therefore $\mathcal{D}$ will always measure 2 identical bits, hence always outputting 0.

On the other hand, let us consider the case where the distinguisher is interacting with the real world. Since the simulator is black-box according to definition given in Theorem 3.3, and since Π expects no input from any party, we can assume, without loss of generality, that the simulator starts by running $\{\pi_i\}_{i\in\{0,\mathtt{S}\}}\mathcal{R}$, forwarding all messages between the malicious interfaces of $\mathcal{R}$ and the environment, and obtaining a state ρ^0 outputted by π_0. Moreover, since $\mathcal{D}$ is honestly running π_1, we know, by the correctness of the protocol, that ρ^0 is half of a Bell-state shared with $\mathcal{D}$. Similarly, since in the ideal world, the protocol never aborts, without loss of generality we can assume that the simulator also sends $c_1 = \top$ to $\mathcal{V}_{|G\rangle}$ when starting (otherwise, we can always convert any simulator that sets $c_1 = \perp$ into a better simulator that sets $c_1 = \top$): the simulator will then receiving a state ρ^σ from $\mathcal{V}_{|G\rangle}$, where ρ^σ is half of a Bell pair shared with the distinguisher. Then, we can call $U^\sigma(\rho^0, \left|0^l\right\rangle, \rho^\sigma)$ the rest of the quantum map performed by the simulator after receiving ρ^0 and ρ^σ, where $\left|0^l\right\rangle$ is an arbitrary auxiliary register. Therefore, the global state obtained when the simulator has finished to run is:

$$(I_2 \otimes U^\sigma \otimes I_2)(|00\rangle + |11\rangle) \otimes \left|0^l\right\rangle \otimes (|00\rangle + |11\rangle) \tag{48}$$

where I_2 is the identity acting on a single qubit. If we consider now the view of the distinguisher right before performing its measurement, we can trace out the map performed by the simulator:

$$\mathrm{Tr}_\sigma((I_2 \otimes U^\sigma \otimes I_2)(|00\rangle + |11\rangle) \otimes \left|0^l\right\rangle \otimes (|00\rangle + |11\rangle)(\langle 00| + \langle 11|) \otimes \left\langle 0^l\right| \otimes (\langle 00| + \langle 11|)(I_2 \otimes U^{\sigma\dagger} \otimes I_2)) \tag{49}$$

By the non-signaling principle, U^σ cannot modify this state since it is traced out. As a consequence, this state is equal to:

$$\mathrm{Tr}_\sigma((I_2 \otimes I_{2^{l+2}} \otimes I_2)(|00\rangle + |11\rangle) \otimes \left|0^l\right\rangle \otimes (|00\rangle + |11\rangle)(\langle 00| + \langle 11|) \otimes \left\langle 0^l\right| \otimes (\langle 00| + \langle 11|)(I_2 \otimes I_{2^{l+2}} \otimes I_2)) \tag{50}$$

But it is easy to see that this state is equal to the identity density matrix of 2 qubits, as we discard, twice, one share of a Bell pair. Hence, after measuring this state in the computational basis, the distinguisher will obtain 0 with probability $1/2$ instead of 1, so the advantage in distinguishing is $1/2 > \varepsilon$. Therefore, Π cannot ε realize $\mathcal{V}_{|G\rangle}$, raising a contradiction. □

Theorem 3.6 (Any graph state is mergeable)**.** *For any graph $G = (V, E)$, $|G\rangle$ is mergeable (Definition 3.5) with respect to the maps $\{\xi_H\}_{H\subseteq[n]}$ that take two lists of X and Z corrections $(x, z) \in (\mathbb{Z}_2^{|H|})^2$ and applies $\mathbf{X}^x\mathbf{Z}^z$ on the input qubits.*

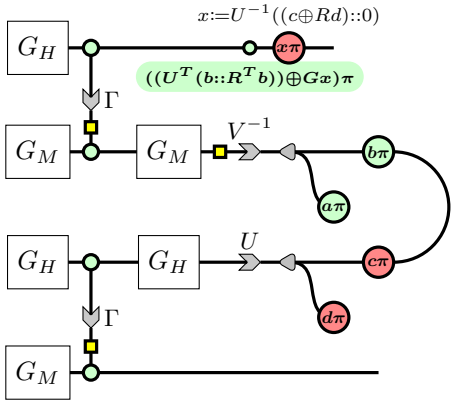
The merge procedure is described diagrammatically in Fig. 6, and for completeness we reformulate it here. Let $n = |V|$, H and M be any partition of V. For simplicity, we assume that we reorder elements of

V to have elements of H ordered before elements of M. Let Γ be the biadjency graph between H and M (cf. Definition A.4). Then, we define ξ_σ as follows (cf. illustration Fig. 6), where the i-th qubit of $|G\rangle$ belongs to register H (resp. M) iff $i \in H$ (resp. M).

- *It applies $\wedge\mathbf{Z}$ gates on any pair (i,j) of qubits of register H iff $(i,j) \in G_H$ and similarly it applies $\wedge\mathbf{Z}$ gates on any pair (i,j) of qubits of register M iff $(i,j) \in G_M$.*
- *It applies Hadamard gates on all qubits of register M.*
- *It computes U, V, r and R according to Lemma A.11 and applies the unitary $|x\rangle \mapsto \left|V^{-1}x\right\rangle$ on register M and $|x\rangle \mapsto |Ux\rangle$ on register H. This is always possible since U and V are invertible. We propose moreover in Lemma A.12 a way to implement them more efficiently, without auxiliary qubits and using only* **CNOT** *and swap operations.*
- *It performs r Bell measurements (projection on one of the four Bell states) between the first r qubits of each register. The Bell measurements are between the i-th qubit of register M with the i-th qubit of register H, where a measurement outcome $(b_i, c_i) \in \{0,1\}^2$ means that the i-th pair was projected on the Bell state $|0c_i\rangle + (-1)^{b_i}|1\bar{c_i}\rangle$. The outcomes are gathered into two vector $b = (b_i)_{i\in[r]}$ and $c = (c_i)_{i\in[r]}$.*
- *It performs a measurement in the $\{H|a_i\rangle\}_{a_i\in\{0,1\}}$ basis on the $|M|-r$ remaining qubits of register M (the outcomes are gathered into a vector a), and a measurement in the computational basis $\{|d_i\rangle\}_{d_i\in\{0,1\}}$ on the $|H|-r$ remaining qubits of register H (the outcomes are gathered into a vector d).*
- *It computes $x \coloneqq U^{-1}\begin{bmatrix} c \oplus Rd \\ \mathbf{0} \end{bmatrix}$, and $z \coloneqq \left(U^T \begin{bmatrix} b \\ R^T b \end{bmatrix}\right) \oplus Gx$ where Gx is the set of neighbours of x as defined in Lemma A.7, and returns the corrections (x,z).*

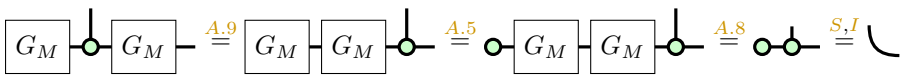
Proof of Theorem 3.6. First, we can decompose the graphs given as input to the merge procedure given in Fig. 6 as described in Lemma A.10 (note that the outcome state might depend on the measurement outcomes a, b, c, d, hence the notation):

$|\psi_{a,b,c,d}\rangle \coloneqq$



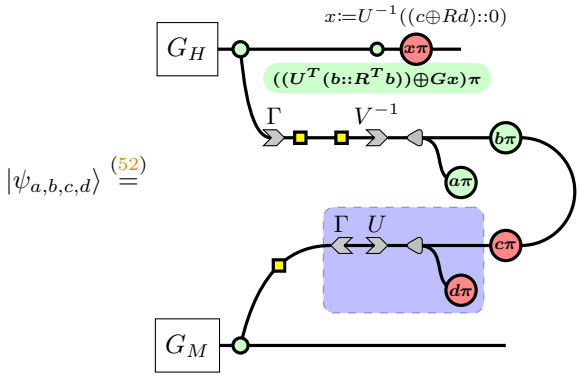
(51)

Then, we can see that:



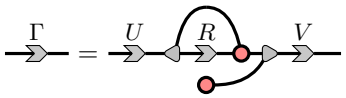
(52)

The same is of course true for G_H instead of G_M, we can therefore simplify the above graph as follows:



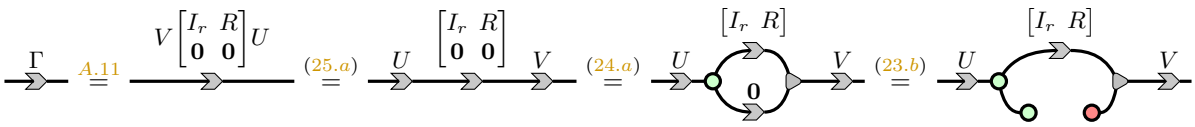
$$|\psi_{a,b,c,d}\rangle \overset{(52)}{=} \tag{53}$$

We will first simplify the colored part, but we prove before an equivalent form of Γ that will also be useful later: Let U, V, r and R be like in Lemma A.11. We claim that:

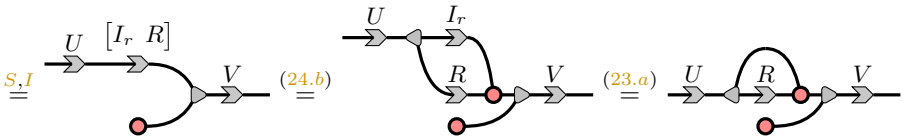


(54)

This is easy to see by mechanically using the diagrammatic representation of block, identity, and zero matrices:

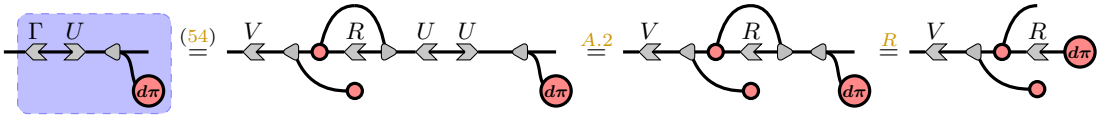


(55)

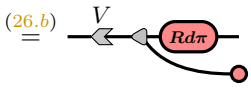


(56)

Therefore, we have:

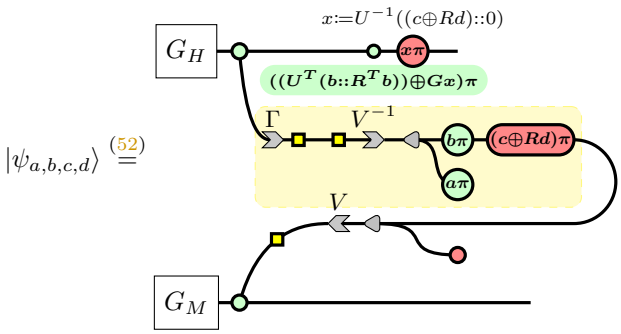


(57)



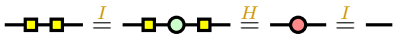
(58)

By injecting this into the colored part of Eq. (53), we get:



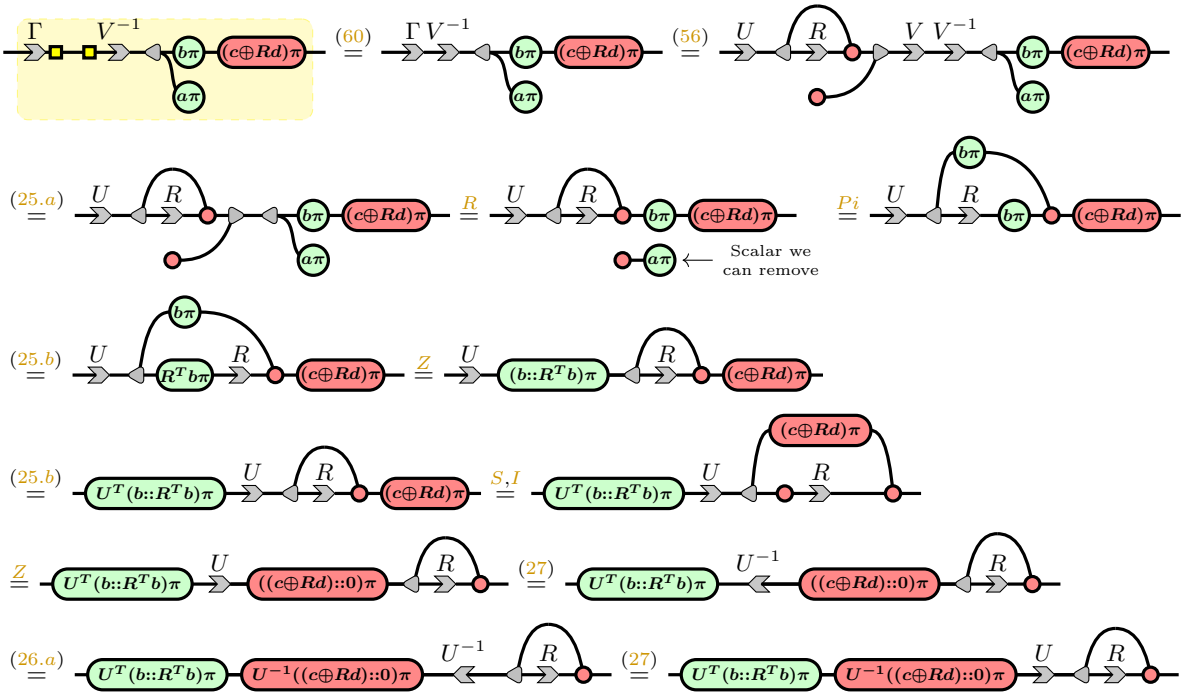
$$|\psi_{a,b,c,d}\rangle \overset{(52)}{=} \tag{59}$$

We focus now on the yellow-colored part of this diagram. First, we can use the well-known equality that is easy to prove as follows:



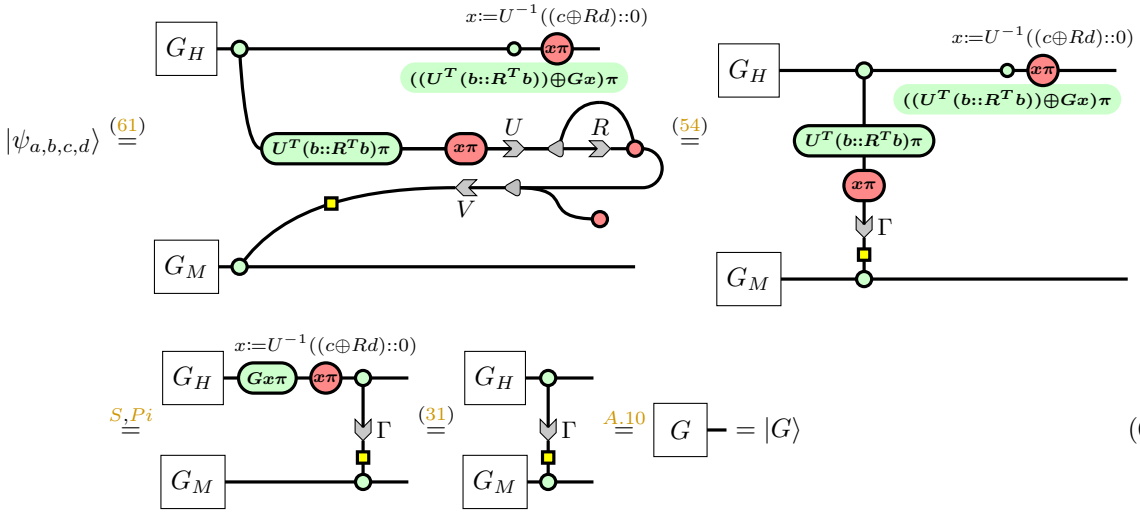
(60)

which gives:



(61)

We inject this back into the yellow-colored part of Eq. (59), after defining $x := U^{-1}((c \oplus Rd)::0)$:



(62)

Which concludes the proof. □

Corollary 3.7 (GHZ states are mergeable)**.** *Any* GHZ *state of size* $|n|$ *(each qubit being a separate register) is mergeable (Definition 3.5) with respect to the collection of quantum maps* $\{\xi_H\}_{H \subseteq [n]}$*, where* ξ_H *takes*

two bits $(x, z) \in \{0,1\}^2$ *as input, applies* $\mathbf{Z}^z$ *of the first qubit, and* $\mathbf{X}^x$ *on all its input qubits (if* $|H|$ *is empty, it does not do anything).*

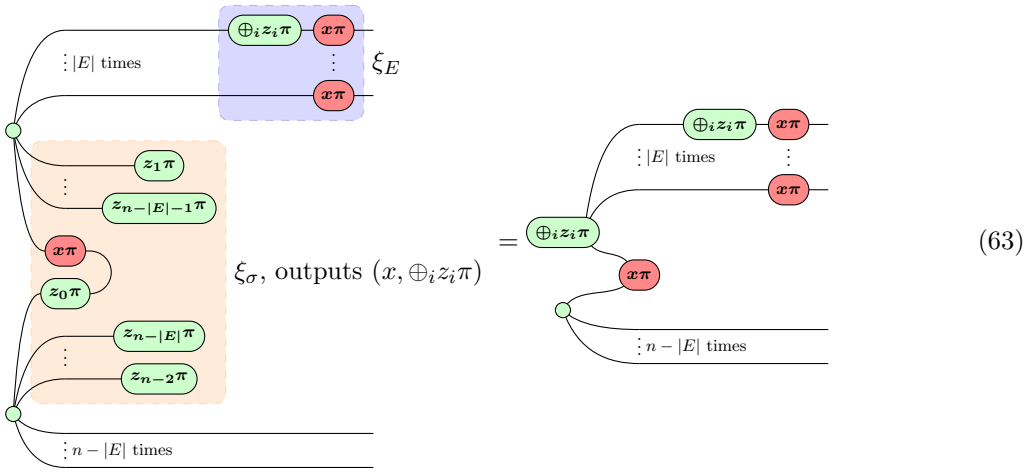
Proof of Corollary 3.7. While this is a direct consequence of Theorem 3.6, we provide here a simpler proof specific to GHZ states. First, we note that if $|H|$ is empty, Eq. (16) is trivially achieved by defining ξ_σ as the map that measures completely its input state, and outputs, say, $x = 0$ and $z = 0$. Since neither ξ_σ nor ξ_H touch the second state at all, this outputs the second state hence achieving Eq. (16).

If $H = [n]$, on the other hand, ξ_σ has only access to the second state. If we define ξ_σ that measures completely its input state, and outputs $x = 0$, $z = 0$ to ξ_H, then ξ_H will left the first state intact, therefore achieving Eq. (16).

If $|H|$ is not empty, then we define ξ_σ as follows:

- ξ_σ performs a Bell measurement (i.e. a projection[11] on one of the four Bell states $|0x\rangle + (-1)^{z_0} |1\bar{x}\rangle$, where $(x, z_0) \in \{0,1\}^2$) between the last qubit of the registers in $R_{\bar{H}}$, and the first qubit of R_H, getting outcomes (x, z_0).
- Then ξ_σ measures all remaining qubits in the Hadamard basis, getting outcomes $\{z_i\}_{i\in\{1,\dots,n-2\}}$
- Finally, ξ_σ outputs x and $z \coloneqq \oplus_{i\in\{0,n-2\}} z_i$.

We prove now that this ξ_σ quantum maps achieves Eq. (16) using ZX calculus:



(63)

(64)

□

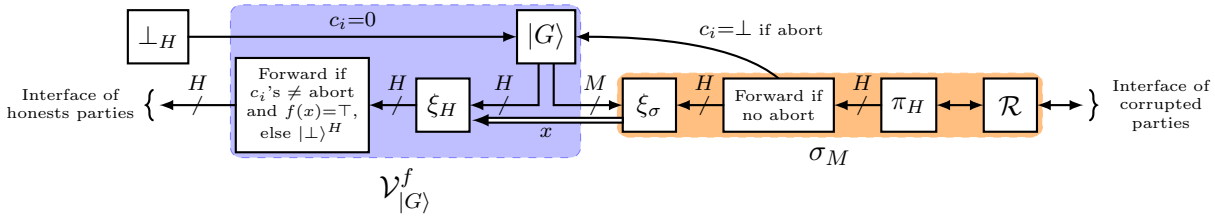
Lemma 3.8 (Security proof of Theorem 3.2)**.** *Let* $M \subseteq [n] \cup \{\mathtt{S}\}$ *be an arbitrary subset of corrupted parties, and* $H = [n] \cup \{\mathtt{S}\} \setminus M$ *be the honest parties. Then there exists a simulator* σ_M *such that* $\pi_H \mathcal{R} \approx \bot_H \mathcal{V}^f_{|G\rangle} \sigma_M$ *for each possible subset* M*.*

[11] This measurement can be done via a CNOT gate, and measuring the first qubit in the Hadamard basis to get z_0 and the second qubit in the computational basis to get x.

Proof of Lemma 3.8. Let $M \subseteq [n] \cup \{\mathtt{S}\}$ be the subset of corrupted parties, and $H = [n] \cup \{\mathtt{S}\} \setminus M$ be the honest parties. We need to prove that there exists a simulator σ_M such that $\pi_H \mathcal{R} \approx \perp_H \mathcal{V}^f_{|G\rangle} \sigma_M$ for each possible subset M.

Remark B.1. *Note that for simplicity, we consider σ_M to be global (i.e. it is a single entity able to communicate will all interfaces), while the constructive cryptography framework typically expect simulators to be local*[12]*: this is without loss of generality since we can easily turn it back into a set of local simulators, one simulator performing the operations of σ_M, where the input (resp. outputs) are obtained (resp. forwarded) from (resp. to) the right interface by mean of the channel $\mathcal{C}$ included in $\mathcal{V}^f_{|G\rangle}$, using the appropriate simulator connected to this interface as a proxy.*

Since it is the most general case, we will focus on proving security for an arbitrary coalition of dishonest parties and source (Fig. 7c). The other cases follow directly from this one. The simulator σ_M can informally be summarized as follows:



(65)

More formally, σ_M is defined as:

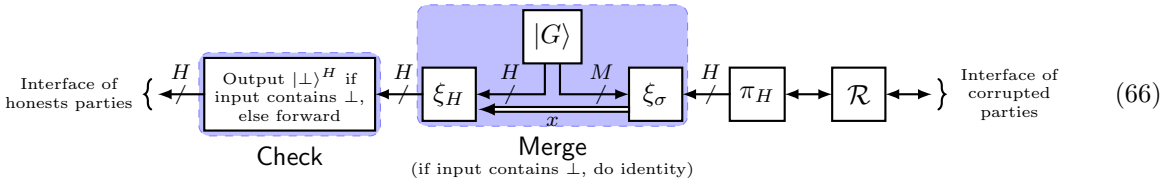
1. σ_M will first locally simulate $\pi_H \mathcal{R}$, where the interfaces of the malicious parties are directly connected to the distinguisher.
2. If an honest party in the local simulation π_H aborts within σ_M, the simulator will send $\perp$ to the ideal functionality and abort. Otherwise, we denote as ρ^{R_H} the state outputted at the end of the protocol by the honest parties in $H \setminus \{\mathtt{S}\}$ (if this set is empty, ρ^{R_H} is just the empty state).
3. Then, the simulator will send $c_i = 1$ to the functionality for all $i \in M \setminus \{\mathtt{S}\}$ to obtain a quantum state $|G\rangle^{R_M}$ (similarly, if $M \setminus \{\mathtt{S}\}$ is empty, the simulator does not send any message and defines $|G\rangle^{R_M}$ as the empty state).
4. The simulator runs the merging map ξ_σ described in Theorem 3.6 applied to ρ^{R_H} and $|G\rangle^{R_M}$ (notice that $|G\rangle^{R_H}$ is replaced with ρ^{R_H}), and gets $(x, y) \in (\{0,1\}^{|H|})^2$.
5. The simulator sends (x, y) on an arbitrary malicious interface in $M \setminus \{\mathtt{S}\}$ (if $M \setminus \{\mathtt{S}\}$ is empty, i.e. only the source may be malicious, then the simulator does not do anything at that step).

We will prove now that $\pi_H \mathcal{R} \approx_{\epsilon'} \perp_H \mathcal{V}^f_{|G\rangle} \sigma_M$ for some ϵ' defined later, by defining a series of hybrid systems close to each others. We recall that $\pi_H \mathcal{R}$ and $\perp_H \mathcal{V}^f_{|G\rangle} \sigma_M$ are the resources corresponding respectively to the concrete protocol and ideal protocol with a subset H of honest parties and $M = \bar{H}$ of dishonest parties.

The first step will be to simplify the system combining the functionality and simulator. We first remark that verifying if $f(x) = \top$ is useless since ξ_σ always output x such that $f(x) = \top$ (we defined f exactly to have this property). The second remark we can make is that it is much simpler to analyse this system if we group the operations differently in order to remove any interaction between the functionality and the simulator. So informally, we will group the operations related to the merging operation together

[12] Note that some framework do not make that choice, like in universal composability.

instead of having them shared between the simulator and the functionality, while we will check if we need to abort only at the very end, giving this informal picture:



(66)

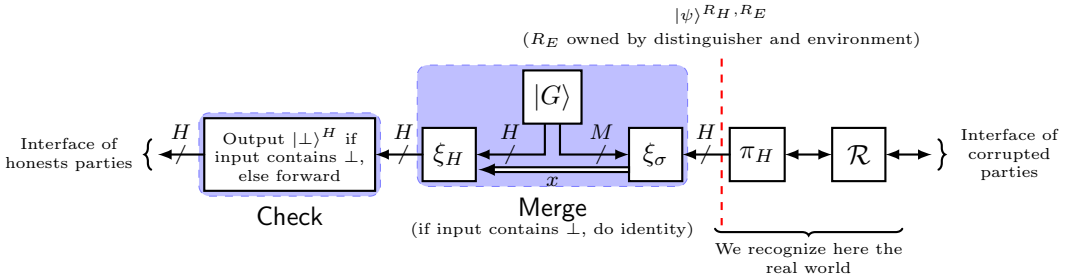
More formally, let Merge be the map working on the registers R_H outputted by the honest parties that applies the identity if one of its input qubit is $|\perp\rangle$ (this can be done via a partial measurement like in the step 2 of the definition of the simulator) and that applies otherwise the merging operation $\xi_H(I_{|H|} \otimes \xi_\sigma)(|G\rangle^{R_H,R_{\bar{H}}} \otimes I_{|H|})$ that merges the state in R_H with a newly created state $|G\rangle$ after applying the corrections. With this definition, the system representing the simulator and the functionality and filters is strictly indistinguishable from this system:

- First we simulate locally $\pi_H\mathcal{R}$ with the distinguisher's input to obtain ρ_H on register R_H.
- Then, we apply Merge.
- Then, we do a partial measurement of the state obtained at the previous step, checking if one input is $|\perp\rangle$. If yes, we replace the state with $|\perp\rangle^{\otimes|H|}$, otherwise we apply the identity (we call this operation Check).
- Finally, for each $i \in H \setminus \{\mathtt{S}\}$ we send the i-th qubit to the party i.

This hybrid system is obviously indistinguishable from $\perp_H \mathcal{V}^f_{|G\rangle}\sigma_M$ as the system is only an identical simplification of the original system (again, $f(x)$ is always equal to $\top$, and otherwise in both cases we abort if any input contains $|\perp\rangle$ and apply the merging operation otherwise).

We remark now that it is nearly equal to $\pi_H\mathcal{R}$, except for the application of Merge and Check at the end. To prove that these two operations do not significantly change the state obtained after the first step, we consider the quantum state obtained at the end of the first step, i.e. when getting the outcomes of π_H. Without loss of generality, we can assume that we run a purified version of the distinguisher and protocol, or, equivalently, that we consider the averaged state (averaging over all randomness involved in the protocol and in the distinguisher) where the purification of the density state is kept on a register kept by the environment[13]. We can also show that the dimension of the purified system is at least $2(n+1)$ (if needed we can add $|0\rangle$ states on the distinguisher's state, this dimension is just chosen large enough so that any later purification has at most this dimension). We call $|\psi\rangle^{R_\sigma,R_E}$ the joint state between the simulator and the environment and distinguisher after receiving the outputs of π_H:

[13]This purification is only used to simplify computations as this way we do not need to consider each run separately. Note that this cannot decrease the distinguishing probability of the distinguisher to purify the protocol, since we can purify any protocol and distinguisher by simply replacing any sampling operation by a measurement of a $|+\rangle$, and any measurement in the computational basis can be replaced by a **CNOT** on an auxiliary qubit kept by the environment, where the target state will then contain the result of the measurement. One can easily see that this is equivalent since we can consider that the state given to the environment is traced out.



(67)

Since $|\psi\rangle^{R_H,R_E}$ is the state obtained in the real world, and because all the operations performed by the distinguisher are the same in both the ideal and the concrete worlds, it is enough to prove that $\mathrm{TD}(((\mathsf{Check} \circ \mathsf{Merge})^{R_H} \otimes I^{R_E}) |\psi\rangle^{R_H,R_E}), |\psi\rangle^{R_H,R_E})$ is small to conclude that the distinguisher[14] could otherwise distinguish two states close in trace distance which is impossible by the laws of physics. First, let $\rho \coloneqq \mathrm{Tr}_E(|\psi\rangle^{R_H,R_E})$. Then, since Π is secure (Theorem 3.2), by assumption there exists p such that:

$$F(\rho, \sigma) \geq 1 - \varepsilon(\lambda) \tag{68}$$

where σ is defined like in Definition 3.1. However, we can remark that σ admits the following purification:

$$|G_p\rangle \coloneqq \sqrt{p}\,|G\rangle\,|0\rangle\left|0^k\right\rangle + \sqrt{(1-p)}\,|\perp^n\rangle\,|1\rangle\left|0^k\right\rangle \tag{69}$$

where k is chosen such that the dimension of $|G_p\rangle$ equals the dimension of $\left|\psi^{R_H,R_E}\right\rangle$, since $\sigma = \mathrm{Tr}_{[n]\setminus H}\,|G_p\rangle\langle G_p|$.

But, by Uhlmann's theorem (see, e.g. [NC10, Thm. 9.4]), there exists two purifications $|\phi_\sigma\rangle$ and $|\phi_\rho\rangle$ of, respectively, σ and ρ, such that

$$F(\rho, \sigma) = |\langle\phi_\sigma \mid \phi_\rho\rangle| \tag{70}$$

Without loss of generality, we can append $|0\rangle$'s to $|\phi_\sigma\rangle$ and $|\phi_\rho\rangle$ to ensure their dimension is equal to those of $|\psi\rangle^{R_H,R_E}$ while maintaining the fact that $F(\rho,\sigma) = |\langle\phi_\sigma \mid \phi_\rho\rangle|$. Moreover, because $|\phi_\rho\rangle$ and $|\psi\rangle^{R_H,R_E}$ (resp. $|\phi_\sigma\rangle$ and $|G_p\rangle$) have the same reduced density matrix[15], there exists U_ρ (resp. U_σ) such that $|\psi\rangle^{R_H,R_E} = (I^{R_H} \otimes U_\rho)\,|\phi_\rho\rangle$ (resp. $|G_p\rangle = (I \otimes U_\sigma)\,|\phi_\sigma\rangle$). Therefore:

$$\mathrm{TD}(((\mathsf{Check} \circ \mathsf{Merge})^{R_H} \otimes I^{R_E})\,|\psi\rangle^{R_H,R_E}, |\psi\rangle^{R_H,R_E}) \tag{71}$$

$$= \mathrm{TD}(((\mathsf{Check} \circ \mathsf{Merge})^{R_H} \otimes U_\rho^{R_E})\,|\phi_\rho\rangle, (I^{R_H} \otimes U_\rho^{R_E})\,|\phi_\rho\rangle) \tag{72}$$

Then, using the triangle inequality twice we get:

$$\begin{aligned} &\leq \mathrm{TD}(((\mathsf{Check} \circ \mathsf{Merge})^{R_H} \otimes U_\rho^{R_E})\,|\phi_\rho\rangle, ((\mathsf{Check} \circ \mathsf{Merge})^{R_H} \otimes U_\rho^{R_E})\,|\phi_\sigma\rangle) \\ &\quad + \mathrm{TD}(((\mathsf{Check} \circ \mathsf{Merge})^{R_H} \otimes U_\rho^{R_E})\,|\phi_\sigma\rangle, (I^{R_H} \otimes U_\rho^{R_E})\,|\phi_\sigma\rangle) \\ &\quad + \mathrm{TD}((I^{R_H} \otimes U_\rho^{R_E})\,|\phi_\sigma\rangle, (I^{R_H} \otimes U_\rho^{R_E})\,|\phi_\rho\rangle) \end{aligned} \tag{73}$$

Then, using the fact that TD is symmetric, cannot be increased with post-processing, and is left unchanged when adding/removing/replacing a unitary on both inner terms (needed to replace U_ρ with U_σ), we can simplify it as:

$$\leq 2\,\mathrm{TD}(|\phi_\rho\rangle, |\phi_\sigma\rangle) + \mathrm{TD}(((\mathsf{Check} \circ \mathsf{Merge})^{R_H} \otimes U_\sigma^{R_E})\,|\phi_\sigma\rangle, (I^{R_H} \otimes U_\sigma^{R_E})\,|\phi_\sigma\rangle) \tag{74}$$

$$= 2\,\mathrm{TD}(|\phi_\rho\rangle, |\phi_\sigma\rangle) + \mathrm{TD}(((\mathsf{Check} \circ \mathsf{Merge})^{R_H} \otimes I^{R_E})\,|G_p\rangle, |G_p\rangle) \tag{75}$$

[14] To be more precise, we could use the second part of the distinguisher that runs after obtaining $|\psi\rangle^{R_H,R_E}$ to distinguish these two states.

[15] It is a standard exercise to show, using the Schmidt decomposition, that two states with the same dimension and the same purification are equal up to a unitary applied on the purification register.

However, by the definition of the trace distance on pure state and by definition of $|\phi_\rho\rangle$ and $|\phi_\sigma\rangle$, we have:

$$\mathrm{TD}(|\phi_\rho\rangle, |\phi_\sigma\rangle) = \sqrt{1 - |\langle\phi_\rho \mid \phi_\sigma\rangle|^2} \overset{(70)}{=} \sqrt{1 - F(\rho,\sigma)^2} \overset{(68)}{\leq} \sqrt{1 - |1-\varepsilon|^2} = \sqrt{1 - (1 + \varepsilon^2 - 2\varepsilon)} \tag{76}$$

$$= \sqrt{2\varepsilon - \varepsilon^2} \tag{77}$$

Moreover, we claim that $((\mathsf{Check} \circ \mathsf{Merge})^{R_H} \otimes I^{R_E}) |G_p\rangle = |G_p\rangle$ and that therefore:

$$\mathrm{TD}(((\mathsf{Check} \circ \mathsf{Merge})^{R_H} \otimes I^{R_E}) |G_p\rangle, |G_p\rangle) = 0 \tag{78}$$

This can be seen by starting from the definition of $|G_p\rangle$ (the proof of this claim ends at Eq. (81)):

$$((\mathsf{Check} \circ \mathsf{Merge})^{R_H} \otimes I^{R_E}) |G_p\rangle = ((\mathsf{Check} \circ \mathsf{Merge})^{R_H} \otimes I^{R_E})(\sqrt{p}\,|G\rangle\,|0\rangle\,|0^k\rangle + \sqrt{1-p}\,|\perp^n\rangle\,|1\rangle\,|0^k\rangle) \tag{79}$$

Then, since Merge behave as identity when the state contains $\perp$, and similarly Check is identity if the state does not contain $\perp$, this can be rewritten as:

$$((\mathsf{Check} \circ \mathsf{Merge})^{R_H} \otimes I^{R_E}) |G_p\rangle = \sqrt{p}(\mathsf{Merge}^{R_H} \otimes I^{R_E}) |G\rangle\,|0\rangle\,|0^k\rangle + \sqrt{1-p}(\mathsf{Check}^{R_H} \otimes I^{R_E}) |\perp^n\rangle\,|1\rangle\,|0^k\rangle \tag{80}$$

Finally $\mathsf{Check}\,|\perp^{|H|}\rangle = |\perp^{|H|}\rangle$ and since any graph state is mergeable (Definition 3.5 and Theorem 3.6), we have:

$$((\mathsf{Check} \circ \mathsf{Merge})^{R_H} \otimes I^{R_E}) |G_p\rangle = \sqrt{p}\,|G\rangle\,|0\rangle\,|0^k\rangle + \sqrt{1-p}\,|\perp^n\rangle\,|1\rangle\,|0^k\rangle = |G_p\rangle \tag{81}$$

finishing the proof of our claim $((\mathsf{Check} \circ \mathsf{Merge})^{R_H} \otimes I^{R_E}) |G_p\rangle = |G_p\rangle$.

Now, we combine Eq. (78) and Eq. (77) into Eq. (75) to get:

$$\mathrm{TD}(((\mathsf{Check} \circ \mathsf{Merge})^{R_H} \otimes I^{R_E}) |\psi\rangle^{R_H,R_E}, |\psi\rangle^{R_H,R_E}) \leq 2\sqrt{2\varepsilon - \varepsilon^2} \tag{82}$$

As discussed previously, the advantage in distinguishing these two states directly gives an upper bound on the advantage of distinguishing the ideal world from the real world (since these two states are actually the states held by the distinguisher at the end of its interaction with the ideal resource or the real protocol). But it is a well known fact that the best probability of distinguishing $|\phi\rangle$ from $|\psi\rangle$ is $\frac{1}{2}(1 + \mathrm{TD}(|\phi\rangle, |\psi\rangle))$, i.e. the advantage in distinguishing $|\psi\rangle$ from $|\phi\rangle$ is actually $\mathrm{TD}(|\phi\rangle, |\psi\rangle)$, so if we define:

$$\varepsilon' \coloneqq 2\sqrt{2\varepsilon - \varepsilon^2} \tag{83}$$

then we have $\pi_H \mathcal{R} \approx_{\varepsilon'} \perp_H \mathcal{V}^f_{|G\rangle} \sigma_M$. For any subset of corrupted party M, we are able to construct such a simulator σ_M. Following Definition 2.1, Π ε'-realizes the functionality $\mathcal{V}^f_{|G\rangle}$, concluding our proof. □

B.2 Proofs of Section 4

Theorem 4.1. *Protocol 4.1 realizes* $\mathcal{V}_{|G\rangle}$.

Proof of Theorem 4.1.
As before, we will follow the Security Definition 2.1 and find simulators for each possible dishonest behaviour. We show the general case in Fig. 10.

Case 1: Correctness. We first need to prove correctness, i.e. $\perp_{[n]\cup\mathsf{S}} \mathcal{V}_{|G\rangle} \approx \tau_{[n]} \perp_{[n]} (\mathcal{V}^f_{|G\rangle} \| \mathcal{R}_{\mathsf{CoinFlip}}) \perp_{\mathsf{S}}$. If all parties are honest, neither $\mathcal{V}^f_{|G\rangle}$ nor $\mathcal{R}_{\mathsf{CoinFlip}}$ will abort, so all parties will receive a part of $|G\rangle$ with the same x. Then, since each party i applies $\mathbf{X}^{x_i}\mathbf{Z}^{(Gx)_i}$, i.e. a part of a stabilizer, the overall resulting state is $\mathbf{X}^x\mathbf{Z}^{Gx}\,|G\rangle \overset{A.7}{=} |G\rangle$. This is exactly the state obtained in $\perp_{[n]\cup\mathsf{S}} \mathcal{V}_{|G\rangle}$. Hence, there exists no distinguisher able to differentiates the two resources, concluding the proof of correctness.

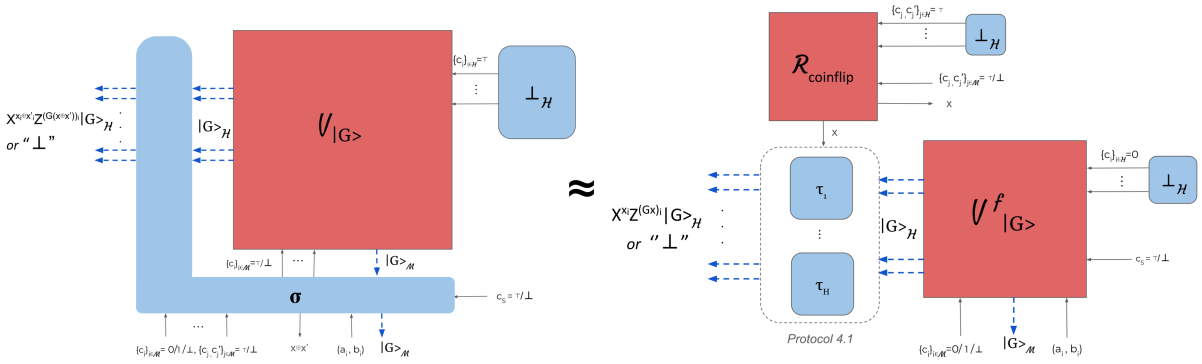


Figure 10: General case of the security proof of Theorem 4.1

Case 2: Security. To prove the security of the protocol, we want to show that for any partition (H, M) of $[n] \cup \{\mathsf{S}\}$, there exists a simulator σ_M such that $\perp_H \sigma_M \mathcal{V}_{|G\rangle} \approx \tau_H \perp_H (\mathcal{V}^f_{|G\rangle} \| \mathcal{R}_{\mathsf{CoinFlip}})$. To match the usual Abstract cryptography terminology, we will denote the left-hand side $\perp_H \sigma \mathcal{V}_{|G\rangle}$ as the ideal world and the right-hand side $\tau_H \perp_H (\mathcal{V}^f_{|G\rangle} \| \mathcal{R}_{\mathsf{CoinFlip}})$ as the real world. As before, using the classical channel allowing communication between malicious parties in $\mathcal{V}^f_{|G\rangle}$, we can consider all local simulators as a single simulator. Let us define the simulator σ formally as in Protocol B.1.

By construction, this simulator follows exactly the abort pattern of the real world protocol, and the exchanged messages are identical. So without loss of generality, we can assume that the distinguisher never aborts[16], and in particular sends valid corrections. Similarly, we can assume that the distinguisher is always sending $c_i = 1$ since the case $c_i = 0$ can easily be simulated by the distinguisher given the transcript for $c_i = 1$ since it contains more information. Under this assumption, both the real and ideal worlds can significantly be simplified:

- In the real world, the distinguisher receives first $|G\rangle_M$, then send some (valid) corrections defining $x'_H = \oplus_i a_i$ and $z'_H := \oplus_i b_i$, and finally receives the remaining qubits $\mathbf{X}^{x_H \oplus x'_H} \mathbf{Z}^{z'_H \oplus (Gx)_H} |G\rangle_H$ together with x that was sampled uniformly at random.
- In the ideal world, the distinguisher receives first $\mathbf{X}^{x_M} \mathbf{Z}^{(Gx)_M} |G\rangle_M$ where x is sampled uniformly at random, then it sends some (valid) corrections defining $x'_H = \oplus_i a_i$ and $z'_H := \oplus_i b_i$, and finally receives the remaining qubits $|G\rangle_H$ together with $x \oplus x'$ where $x'_M := (V^T)^{-1} \begin{bmatrix} b \\ \mathbf{0} \end{bmatrix}$, and $x' := \begin{bmatrix} x'_H \\ x'_M \end{bmatrix}$.

First, we can realize that since $\mathbf{X}^{x_M} \mathbf{Z}^{(Gx)_M} |G\rangle = \mathbf{X}^{x_M} \mathbf{Z}^{(Gx)_M} (\mathbf{X}^x \mathbf{Z}^{Gx} |G\rangle) = \mathbf{X}^{x_H} \mathbf{Z}^{(Gx)_H} |G\rangle$, the ideal world is indistinguishable from a world where the distinguisher receives $|G\rangle_M$ while the functionality applies $\mathbf{X}^{x_H} \mathbf{Z}^{(Gx)_H}$ on the remaining qubits. This gives this equivalent hybrid system:

- The distinguisher receives first $|G\rangle_M$ then it sends some (valid) corrections, defining $x'_H = \oplus_i a_i$ and $z'_H := \oplus_i b_i$, and finally receives the remaining qubits $\mathbf{X}^{x_H} \mathbf{Z}^{(Gx)_H} |G\rangle_H$ (where x is sampled uniformly at random) together with $x \oplus x'$ where $x'_M := (V^T)^{-1} \begin{bmatrix} b \\ \mathbf{0} \end{bmatrix}$, and $x' := \begin{bmatrix} x'_H \\ x'_M \end{bmatrix}$.

Then, we can define $\hat{x} := x \oplus x'$. Since x is independent of x', the probability of sampling any $\hat{x}$ is equal to the probability of sampling any x, we can therefore sample x' instead of x, and use the fact that $x = \hat{x} \oplus x'$. So, the above system can be turned into the following indistinguishable system:

[16] We can always turn a distinguisher that aborts into a distinguisher that does not abort while increasing the probability of distinguishing, for instance by intercepting all abort messages from the original distinguisher, and continuing instead the protocol with arbitrary non-aborting inputs while sending to the original distinguisher the messages that would have been sent in both the ideal and the real protocol.

Protocol B.1 Simulator σ

1. First, σ sends $c_i = \perp$ to $\mathcal{V}_{|G\rangle}$ in order to receive a share $|G\rangle_M$ of $|G\rangle$ for all malicious parties in the set M of corrupted parties.

2. Then, σ receives c_i from each malicious party i (in the real world these messages are sent to $\mathcal{V}^f_{|G\rangle}$ by the distinguisher). If any c_i is equal to $\perp$, then σ also aborts after sending $c'_i = \perp$ to $\mathcal{V}_{|G\rangle}$ to make it abort as well.

3. σ samples a random stabilizer $x \xleftarrow{\$} \{0,1\}^n$, applies $\mathbf{X}^{x_{M'}}\mathbf{Z}^{(G'_M x)_{M'}}$ where $M' \coloneqq \{i \mid c_i = 1\}$ on the received $|G\rangle_M$ and sends the resulting qubits of parties in M' to the distinguisher.

4. Then, σ receives a set of corrections $(a_i, b_i) \in (\{0,1\}^{|H|})^2$ for $i \in M'$, computes $x'_H \coloneqq \oplus_i a_i$ and $z'_H \coloneqq \oplus_i b_i$, and checks that $f_G(M', x'_H, z'_H) = \top$ (otherwise it sends $c'_i = \perp$ to $\mathcal{V}_{|G\rangle}$ for all i and aborts).

5. Since $f_G(M', x'_H, z'_H) = \top$, let b be like in Theorem 3.2, and let us define $x'_M \coloneqq (V^T)^{-1}\begin{bmatrix} b \\ \mathbf{0} \end{bmatrix}$, and $x' \coloneqq \begin{bmatrix} x'_H \\ x'_M \end{bmatrix}$.

6. If the set of corrupted parties contains no party in $[n]$, the simulator can stop. Otherwise, it will receive bits c''_i (sent to $\mathcal{R}_{\mathsf{CoinFlip}}$ in the real world and called c_i there). It will then send $x \oplus x'$ to parties that sent $c''_i = \perp$, wait for a bit from each party (sent to $\mathcal{R}_{\mathsf{CoinFlip}}$ in the real world and denoted c'_i), and abort if one of these party sent $\perp$, by sending $c'_i = \perp$ to the ideal functionality for all $i \in M$. Finally, it broadcasts x to all parties such that $c_i = 1$.

7. For each party i such that $c_i = 0$, σ will apply on the i-th qubit (that remains from the second step) the operation $\mathbf{X}^{x_i \oplus x'_i}\mathbf{Z}^{(G(x \oplus x'))_i}$, and output the resulting qubit to the distinguisher.

8. Finally, it outputs $c'_i = \top$ to $\mathcal{V}_{|G\rangle}$ in order to let the functionality broadcast $|G\rangle_{\bar{M}}$ to all honest parties.

- The distinguisher receives first $|G\rangle_M$ then it sends some (valid) corrections defining $x'_H = \oplus_i a_i$ and $z'_H \coloneqq \oplus_i b_i$, and finally receives the remaining qubits $\mathbf{X}^{\hat{x}_H \oplus x'_H}\mathbf{Z}^{(G(\hat{x}\oplus x'))_H}|G\rangle_H$ (where $\hat{x}$ is sampled uniformly at random) together with $\hat{x}$ where $x'_M \coloneqq (V^T)^{-1}\begin{bmatrix} b \\ \mathbf{0}\end{bmatrix}$, and $x' \coloneqq \begin{bmatrix} x'_H \\ x'_M\end{bmatrix}$.

Finally, we have:

$$(Gx')_H = \begin{bmatrix} G_H & \Gamma^T\end{bmatrix} x' = G_H x'_H \oplus U^T \begin{bmatrix} I_r & 0 \\ R^T & 0\end{bmatrix} V^T x'_M = G_H x'_H \oplus \left(U^T \begin{bmatrix} I_r & 0 \\ R^T & 0\end{bmatrix} V^T\right)\left((V^T)^{-1}\begin{bmatrix} b \\ \mathbf{0}\end{bmatrix}\right) \tag{84}$$

$$= G_H x'_H \oplus U^T \begin{bmatrix} b \\ R^T b\end{bmatrix} \overset{3.2}{=} G_H x'_H \oplus U^T (U^T)^{-1}(z'_H \oplus G_H x'_H) = z'_H \tag{85}$$

Therefore, we get $\mathbf{X}^{\hat{x}_H \oplus x'_H}\mathbf{Z}^{(G(\hat{x}\oplus x'))_H}|G\rangle_H = \mathbf{X}^{\hat{x}_H \oplus x'_H}\mathbf{Z}^{z'_H \oplus (G\hat{x})_H}|G\rangle_H$. Hence, this last system is actually equal to the real world, which concludes the indistinguishability proof. □

Corollary 4.2. *Assuming the existence of a protocol for graph state verification fulfilling properties described in Theorem 3.2 and a coin flipping protocol realizing $\mathcal{R}_{\mathsf{CoinFlip}}$, there exists a protocol realizing $\mathcal{V}_{|G\rangle}$.*

Proof of Corollary 4.2. This is a direct consequence of Theorem 3.2 and Theorem 4.1, where we run in Protocol 4.1 the graph state verification instead of $\mathcal{V}^f_{|G\rangle}$ and the coin flipping protocol instead of $\mathcal{R}_{\mathsf{CoinFlip}}$.

More precisely, in Theorem 3.2, we first proved the equivalence $\pi_{[n]\cup\mathsf{S}}\mathcal{R} \approx \perp_{[n]\cup\mathsf{S}}\mathcal{V}^f_{|G\rangle}$ between a concrete verification protocol and an ideal, but gruesome, resource. Then, in Theorem 4.1, we explicited the protocol $\{\tau_i\}_{i\in[n]}$, that applies a random stabilizer to the output state of the verification protocol, to construct a simpler functionality $\mathcal{V}_{|G\rangle}$ from this gruesome resource alongside with a coin flipping resource. In particular, we proved the equivalence $\perp_{[n]\cup\mathsf{S}}\mathcal{V}_{|G\rangle} \approx \tau_{[n]}\perp_{[n]}(\mathcal{V}^f_{|G\rangle}\|\mathcal{R}_{\mathsf{CoinFlip}})\perp_{\mathsf{S}}$.

Let $\pi'_{[n]}$ be a concrete protocol securely realising the ideal functionality $\mathcal{R}_{\mathsf{CoinFlip}}$ using resource $\mathcal{R}'$. We have, in particular, that $\pi'_{[n]}\mathcal{R}' \approx \perp_{[n]}\mathcal{R}_{\mathsf{CoinFlip}}$. By composability of resources within the AC framework, we get the correctness equivalence :

$$\pi_{[n]\cup\mathsf{S}} \circ \tau_{[n]}(\mathcal{R}\|\pi'_{[n]}\mathcal{R}') \approx \perp_{[n]\cup\mathsf{S}}\mathcal{V}_{|G\rangle}. \tag{86}$$

For all subset of malicious parties M, we can similarly appropriately compose the simulators from the proofs of the secure construction of $\mathcal{R}_{\mathsf{CoinFlip}}$, $\mathcal{V}^f_{|G\rangle}$ and $\mathcal{V}_{|G\rangle}$ to obtain simulators σ_M. We can use these simulators and the composability of AC to similarly prove all the equivalences necessary in the Security Definition 2.1. This proves the secure construction of the ideal functionality $\mathcal{V}_{|G\rangle}$ by a graph state verification protocol fulfilling properties described in Theorem 3.2 and a coin flipping protocol realizing $\mathcal{R}_{\mathsf{CoinFlip}}$. □

B.3 Proofs of Section 5

Lemma 5.2. *Let $\Pi = \{\pi_i\}_{i\in[n]\cup\mathsf{S}}$ be a protocol generating, when all parties are honest, a state $|G\rangle$ shared among all parties but the source. We assume that Π has simultaneous abortion (Remark 5.1), i.e. that for any adversary $\mathcal{A}$, either all honest parties abort at the same time with some probability $1-p$ or accept and output the averaged state $\rho_\top$. Then, if* any *of the following conditions is fulfilled, this protocol is an ε-graph state verification protocol according to Definition 3.1:*

- *If $p(1 - F(\rho_\top, \mathrm{Tr}_{[n]\setminus H}(|G\rangle\langle G|))) \leq \varepsilon$. This denotes the fact that the probability of accepting and outputting a state far from $|G\rangle$ to the honest parties is small.*

- *Or if $F(\rho_\top, \mathrm{Tr}_{[n]\setminus H}(|G\rangle\langle G|)) \geq 1-\varepsilon$. This corresponds to the protocol's property to create a state close to the desired state. Note that this condition is strictly stronger than the first one, since an adversary might be able to produce such a state with negligible probability. Yet, unconditionally secure protocols might prefer this formulation.*

Proof of Lemma 5.2. Since the protocol is correct by assumption, the first point of Definition 3.1 is trivially fulfilled. We focus now on proving that the protocol is secure according to Definition 3.1. Since the protocol has simultaneous abortion, the first part of the security statement is trivially true, hence we just need to find σ such that Eq. (5) is true. Let $\rho \coloneqq \mathrm{Tr}_{[n]\setminus H}(\pi_H \mathcal{R}\mathcal{A})$ be the averaged state obtained at the end of the interaction with $\mathcal{A}$. Then, since π has simultaneous abortion, we know that there exists $p \coloneqq \left\langle \perp^{|H|} \right| \rho \left| \perp^{|H|} \right\rangle$ (actually this is the same p as in the lemma) such that:

$$\rho = (1-p) \left|\perp^{|H|}\right\rangle\!\left\langle\perp^{|H|}\right| + p\rho_\top \tag{87}$$

where $\rho_\top$ is a normalized state orthogonal to $\left|\perp^{|H|}\right\rangle\!\left\langle\perp^{|H|}\right|$, corresponding to the averaged state obtained when no party aborted.

Let us define

$$\sigma \coloneqq p\,\mathrm{Tr}_{[n]\setminus H}(|G\rangle\langle G|) + (1-p)\left|\perp^{|H|}\right\rangle\!\left\langle\perp^{|H|}\right| \tag{88}$$

and find ε such that $F(\rho,\sigma) \geq 1-\varepsilon$, or, equivalently, such that $1-F(\rho,\sigma) \leq \varepsilon$. To compute this quantity, we can first use the strong concavity of fidelity ([NC10, Thm. 9.7]) that states that $F(\sum_i p_i\rho_i, \sum_i q_i\sigma_i) \geq \sum_i \sqrt{p_i q_i} F(\rho_i,\sigma_i)$. Therefore, we have:

$$\begin{aligned} 1-F(\rho,\sigma) &\leq 1-(1-p)F(\left|\perp^{|H|}\right\rangle\!\left\langle\perp^{|H|}\right|, \left|\perp^{|H|}\right\rangle\!\left\langle\perp^{|H|}\right|) - pF(\rho_\top, \mathrm{Tr}_{[n]\setminus H}(|G\rangle\langle G|)) && (89)\\ &= p(1-F(\rho_\top, \mathrm{Tr}_{[n]\setminus H}(|G\rangle\langle G|))) && (90)\end{aligned}$$

where the last equality is a direct consequence of $F(|\phi\rangle\langle\phi|, |\phi\rangle\langle\phi|) = 1$ when $|\phi\rangle$ is a pure state. If any of the first two assumptions of this lemma are fulfilled, we can directly inject it in Eq. (90) to obtain Eq. (5). This shows that the protocol is an ε-graph state verification protocol according to Definition 3.1 (for the second assumption, we upper bound p by 1). □

Lemma 5.3. *If a protocol has simultaneous abortion (Remark 5.1), and if the probability (on the randomness of $\mathcal{A}$ and the whole protocol) to have no abort and a final state far from the target $|G\rangle$ is small, more formally:*

$$\Pr\left[\mathrm{Tr}_{[n]\setminus H}|\psi_i\rangle \neq \left|\perp^H\right\rangle \wedge \sqrt{1-F^2(\mathrm{Tr}_{[n]\setminus H}|\psi_i\rangle\langle\psi_i|, \mathrm{Tr}_{[n]\setminus H}|G\rangle\langle G|)} \geq \eta \;\middle|\; |\psi_i\rangle \leftarrow \pi_H\mathcal{R}\mathcal{A}\right] \leq \delta \tag{18}$$

or, equivalently,

$$\Pr\left[\mathrm{Tr}_{[n]\setminus H}|\psi_i\rangle \neq \left|\perp^H\right\rangle \wedge \min_U \mathrm{TD}((I^H\otimes U^M)|\psi_i\rangle, |G\rangle) \geq \eta \;\middle|\; |\psi_i\rangle \leftarrow \pi_H\mathcal{R}\mathcal{A}\right] \leq \delta \tag{19}$$

then this protocol is an $(\delta+\eta^2)$-graph state verification protocol according to Definition 3.1.

Proof of Lemma 5.3. First, the fact that these two definitions are equivalent comes from the fact that for pure states, $\mathrm{TD}(|\psi\rangle, |\phi\rangle) = \sqrt{1-|\langle\psi|\phi\rangle|^2}$ ([NC10, Eq. 9.99]), hence:

$$\begin{aligned} \min_U \mathrm{TD}((I^H\otimes U^M)|\psi_i\rangle, |G\rangle) &= \min_U \sqrt{1-|\langle G|(I^H\otimes U^M)|\psi_i\rangle|^2} && (91)\\ &= \sqrt{1-\max_U |\langle G|(I^H\otimes U^M)|\psi_i\rangle|^2} && (92)\\ &= \sqrt{1-F^2(\mathrm{Tr}_{[n]\setminus H}|\psi_i\rangle\langle\psi_i|, \mathrm{Tr}_{[n]\setminus H}|G\rangle\langle G|)} && (93)\end{aligned}$$

where the last equality comes from Uhlmann's theorem ([NC10, Ex. 9.15]), and additionally remarking that all purifications are equal up to a local unitary on the purified space. In the following, we will therefore only consider the assumption involving the fidelity.

We will now simplify the assumption Eq. (18), but first let us define some notations. First, it will be handy to denote, like in Lemma 5.2, $\rho_\top$ as the averaged normalized state obtained by honest parties assuming that the protocol has not aborted:

$$\rho_\top \coloneqq \mathop{\mathbb{E}}_{\substack{|\psi_i\rangle \leftarrow \pi_H \mathcal{RA} \\ \mathrm{Tr}_{[n]\setminus H} |\psi_i\rangle \neq \left|\perp^H\right\rangle}} \left[\mathrm{Tr}_{[n]\setminus H} \left|\psi_i\right\rangle\!\left\langle\psi_i\right| \right] \tag{94}$$

If we denote by $\{|\psi_i\rangle\}_i$ the set of all states producible by a $\pi_H \mathcal{RA}$, and if p_i represents the probability of outputting $|\psi_i\rangle$ assuming that it is not aborting ($\mathrm{Tr}_{[n]\setminus H} |\psi_i\rangle \neq \left|\perp^H\right\rangle$), we have therefore:

$$\rho_\top = \mathrm{Tr}_{[n]\setminus H} \left(\sum_i p_i \left|\psi_i\right\rangle\!\left\langle\psi_i\right| \right) \tag{95}$$

To simplify further the notation we define for brevity:

$$F_i \coloneqq F(\mathrm{Tr}_{[n]\setminus H} \left|\psi_i\right\rangle\!\left\langle\psi_i\right|, \mathrm{Tr}_{[n]\setminus H} \left|G\right\rangle\!\left\langle G\right|) \tag{96}$$

$$F \coloneqq F(\mathrm{Tr}_{[n]\setminus H} \left|\psi\right\rangle\!\left\langle\psi\right|, \mathrm{Tr}_{[n]\setminus H} \left|G\right\rangle\!\left\langle G\right|) \tag{97}$$

Using this notation, we can simplify the LHS of the assumption Eq. (18) as:

$$\Pr\left[\mathrm{Tr}_{[n]\setminus H} |\psi\rangle \neq \left|\perp^H\right\rangle \wedge \sqrt{1-F^2} \geq \eta \;\middle|\; |\psi\rangle \leftarrow \pi_H \mathcal{RA} \right] \tag{98}$$

$$= \Pr_{|\psi\rangle \leftarrow \pi_H \mathcal{RA}} \left[\mathrm{Tr}_{[n]\setminus H} |\psi\rangle \neq \left|\perp^H\right\rangle \right] \Pr_{\substack{|\psi\rangle \leftarrow \pi_H \mathcal{RA} \\ \mathrm{Tr}_{[n]\setminus H} |\psi\rangle \neq \left|\perp^H\right\rangle}} \left[\sqrt{1-F^2} \geq \eta \right] \tag{99}$$

By defining $p \coloneqq \Pr_{|\psi\rangle \leftarrow \pi_H \mathcal{RA}} \left[\mathrm{Tr}_{[n]\setminus H} |\psi\rangle \neq \left|\perp^H\right\rangle \right]$ as the probability of non aborting, this is equal to:

$$p \Pr_{\substack{|\psi\rangle \leftarrow \pi_H \mathcal{RA} \\ \mathrm{Tr}_{[n]\setminus H} |\psi\rangle \neq \left|\perp^H\right\rangle}} \left[\sqrt{1-F^2} \geq \eta \right] \tag{100}$$

Then, using the fact that the probability for $|\psi\rangle$ to be equal to $|\psi_i\rangle$ is p_i, we can simplify this as:

$$p \Pr_{\substack{|\psi\rangle \leftarrow \pi_H \mathcal{RA} \\ \mathrm{Tr}_{[n]\setminus H} |\psi\rangle \neq \left|\perp^H\right\rangle}} \left[\sqrt{1-F^2} \geq \eta \right] = p \sum_p p_i \Pr\left[\sqrt{1-F_i^2} \geq \eta \right] = p \sum_{p | \sqrt{1-F_i^2} \geq \eta} p_i \tag{101}$$

Therefore, assumption Eq. (18) is equivalent to:

$$p \sum_{i | \sqrt{1-F_i^2} \geq \eta} p_i \leq \delta \tag{102}$$

Now, we can remark that using Lemma 5.2, it is enough to show that $p(1-F(\rho_\top, \mathrm{Tr}_{[n]\setminus H}(|G\rangle\!\langle G|))) \leq \varepsilon$ to prove that the protocol is an ε-graph state verification protocol. Since we aim to upper bound

$p(1 - F(\rho_\top, \mathrm{Tr}_{[n]\setminus H}(|G\rangle\!\langle G|)))$, we can simplify this expression as follows:

$$
\begin{aligned}
& p(1 - F(\rho_\top, \mathrm{Tr}_{[n]\setminus H}(|G\rangle\!\langle G|))) && (103)\\
& \overset{(95)}{=} p\left(1 - F\left(\sum_i p_i \,\mathrm{Tr}_{[n]\setminus H}\,|\psi_i\rangle\,, \mathrm{Tr}_{[n]\setminus H}(|G\rangle\!\langle G|)\right)\right) && (104)\\
& \leq p\left(1 - \sum_i p_i F_i\right) && \text{(Concavity of fidelity ([NC10, Ex. 9.20]))}\\
& = p\sum_i p_i\,(1 - F_i) && (\textstyle\sum_i p_i = 1)\\
& = p\sum_{i|\sqrt{1-F_i^2}\geq\eta} p_i\,(1 - F_i) + p\sum_{i|\sqrt{1-F_i^2}<\eta} p_i\,(1 - F_i) && \text{(Split the sum)}\\
& \leq \delta + p\sum_{i|\sqrt{1-F_i^2}<\eta} p_i\,(1 - F_i) && \text{(Fidelity is } \geq 0 + \text{Eq. (102))}
\end{aligned}
$$

Now, in the remaining terms, we assume $\sqrt{1 - F_i^2} < \eta$, i.e. $1 - F_i^2 < \eta^2$. But since $0 \leq F_i \leq 1$, we have $0 \leq F_i^2 < F_i$ and therefore $1 - F_i \leq 1 - F_i^2 < \eta^2$. Therefore, after injecting this in the previous equation, we get:

$$
\begin{aligned}
p(1 - F(\rho_\top, \mathrm{Tr}_{[n]\setminus H}(|G\rangle\!\langle G|))) &\leq \delta + p\sum_{i|\sqrt{1-F_i^2}<\eta} p_i\eta^2 && (105)\\
&\leq \delta + \eta^2 && \text{(Upper bound probabilities by 1)}
\end{aligned}
$$

□

Lemma 5.4. *The protocol defined in [PCW+11], assuming that we use a broadcast channel to transmit the abort bit, is an ε-graph state verification protocol for $\varepsilon \coloneqq \frac{1}{2^{S/2}}(4n+1)$, where 2^S is the average number of tests before outputting a state as defined in [PCW+11] and n is the total number of parties. As a result, it $(2\sqrt{2\varepsilon - \varepsilon^2})$-realizes $\mathcal{V}^f_{|G\rangle}$ as defined in Theorem 3.2, and can be turned into a protocol that $(2\sqrt{2\varepsilon - \varepsilon^2})$-realizes $\mathcal{V}_{|G\rangle}$.*

Proof of Lemma 5.4. This is a corollary of Lemma 5.3, using the second version of the assumption (Eq. (19)). The probability in Eq. (19) corresponds exactly to the $\Pr[C_\varepsilon]$ defined in [PCW+11] right before Theorem 3. This same theorem actually states that for any $\varepsilon > 0$, $\Pr[C_\varepsilon] \leq \frac{4n}{2^S h\varepsilon^2}$, where 2^S is the average number of tests before outputting a state as defined in [PCW+11], n is the total number of parties, and h is the number of honest parties.

In AC the distance between two resources is the same irrespective of the number of corrupted parties. We thus aim to upper bound this by a number independent from the number of malicious parties. We can assume that at least one party is honest, i.e. $h \geq 1$, to get $\Pr[C_\varepsilon] \leq \frac{4n}{2^S\varepsilon^2}$. Since this is true for any ε, we can in particular define:

$$
\begin{aligned}
\eta &\coloneqq \sqrt[4]{\frac{1}{2^S}} && (106)\\
\delta &\coloneqq \frac{4n}{2^S\eta^2} = \frac{4n}{2^{S/2}} && (107)
\end{aligned}
$$

and we have $\Pr[C_\eta] \leq \delta$. So we can apply Lemma 5.3 to show that the protocol is a $\delta + \eta^2$-graph state verification protocol, i.e. an ε-graph state verification protocol for $\varepsilon \coloneqq \delta + \eta^2 = \frac{4n}{2^{S/2}} + \sqrt{\frac{1}{2^S}} = \frac{1}{2^{S/2}}(4n+1)$. Finally, we can conclude the proof by using Theorem 3.2 and Corollary 4.2 that directly show that this protocol realizes $\mathcal{V}^f_{|G\rangle}$ and can be turned, with little changes, into a protocol realizing $\mathcal{V}_{|G\rangle}$. □

Lemma 5.5. *We define, as in the theorem 3 of [UM22] (where we use the fact that conditioned on non-aborting, we have $N_{\mathsf{pass}} \geq \lambda J N_{\mathsf{test}} - \frac{N_{\mathsf{test}}}{2J}$ as described in protocol 2 to simplify the expression of p_0 and avoid any dependency on a number that might be different every time we run the protocol):*

- *$J = 2^n$ or $J = n$ depending on G as described in [UM22, Thm. 2],*
- *λ be the security parameter growing polynomially with the number of tests*
- *m and c some positive constants chosen so that p_0 and η_0 defined later are greater than 0,*
- *$p_0 \coloneqq [1 - \sum_{x=0}^{\lambda}(1-\frac{1}{n})^x(\frac{1}{n}J^{\frac{-2cm}{3}})^{\lambda-x}]^J$ (we got rid of the number of honest parties $|H| \geq 1$ since we want this to be independent of the number of malicious parties)*
- *$\eta_0 \coloneqq \left(\frac{1}{\lambda} - \frac{1}{\lambda^2}\right) + \left(1+\frac{1}{\lambda}\right)\frac{\sqrt{c}+1/2}{J}$*

The symmetric protocol 2 defined in [UM22], assuming that we use a broadcast channel to transmit the abort bit, is an ε-graph state verification protocol for $\varepsilon \coloneqq 1 - p_0 + 2\eta_0 - \eta_0^2$. As a result, it $(2\sqrt{2\varepsilon - \varepsilon^2})$-realizes $\mathcal{V}^f_{|G\rangle}$ as defined in Theorem 3.2, and can be turned into a protocol that $(2\sqrt{2\varepsilon-\varepsilon^2})$-realizes $\mathcal{V}_{|G\rangle}$.

Proof of Lemma 5.5. This is mostly a corollary of Lemma 5.3, using the first version of the assumption (Eq. (18)). Indeed, using [UM22, Thm. 3], and by denoting by ρ_i the reduced state outputted by honest parties during a given run (this corresponds to ρ_H^{avg} in [UM22, Thm. 3]), we know that for any adversary $\mathcal{A}$:

$$\Pr\left[\, F(\rho_i, \mathrm{Tr}_{[n]\setminus H}\,|G\rangle\langle G|) \geq 1-\eta_0 \;\middle|\; \rho_i \leftarrow \mathrm{Tr}_{[n]\setminus H}\,\pi_H \mathcal{R}\mathcal{A}, \rho_i \neq |\perp^H\rangle \,\right] \geq p_0 \tag{108}$$

But for any ρ_i,

$$F(\rho_i, \mathrm{Tr}_{[n]\setminus H}\,|G\rangle\langle G|) \geq 1-\eta_0 \Leftrightarrow \sqrt{1-F^2(\rho_i, \mathrm{Tr}_{[n]\setminus H}\,|G\rangle\langle G|)} \leq \sqrt{1-(1-\eta_0)^2} = \sqrt{2\eta_0-\eta_0^2} \tag{109}$$

So let $\eta \coloneqq \sqrt{2\eta_0 - \eta_0^2}$, and $\delta \coloneqq 1 - p_0$. We have therefore:

$$\Pr\left[\, \sqrt{1-F^2(\rho_i, \mathrm{Tr}_{[n]\setminus H}\,|G\rangle\langle G|)} \geq \eta_0 \;\middle|\; \rho_i \leftarrow \mathrm{Tr}_{[n]\setminus H}\,\pi_H \mathcal{R}\mathcal{A}, \rho_i \neq |\perp^H\rangle \,\right] \leq \delta \tag{110}$$

and, since $\Pr[X \wedge E] = \Pr[x \mid E]\Pr[E] \leq \Pr[x \mid E]$, we get as well:

$$\Pr\left[\, \rho_i \neq |\perp^H\rangle \wedge \sqrt{1-F^2(\rho_i, \mathrm{Tr}_{[n]\setminus H}\,|G\rangle\langle G|)} \leq \eta_0 \;\middle|\; \rho_i \leftarrow \mathrm{Tr}_{[n]\setminus H}\,\pi_H \mathcal{R}\mathcal{A} \,\right] \leq \delta \tag{111}$$

So according to Lemma 5.3, the protocol is an $(\delta+\eta^2)$-graph state verification protocol, i.e. an ε-graph state verification protocol for $\varepsilon \coloneqq \delta + \eta^2 = 1 - p_0 + 2\eta_0 - \eta_0^2$. Finally, we can conclude the proof by using Theorem 3.2 and Corollary 4.2 that directly show that this protocol realizes $\mathcal{V}^f_{|G\rangle}$ and can be turned, with little changes, into a protocol realizing $\mathcal{V}_{|G\rangle}$. □