EasyPQC: Verifying Post-Quantum Cryptography

Manuel Barbosa
University of Porto (FCUP) and INESC TEC
Porto, Portugal
mbb@fc.up.pt

Gilles Barthe
MPI-SP and IMDEA Software Institute
Bochum, Germany
gjbarthe@gmail.com

Xiong Fan
Algorand, Inc.
Boston, USA
leofanxiong@gmail.com

Benjamin Grégoire
INRIA
Sophia-Antipolis, France
benjamin.gregoire@inria.fr

Shih-Han Hung
University of Texas
Austin, USA
shung@cs.utexas.edu

Jonathan Katz
University of Maryland
College Park, USA
jkatz2@gmail.com

Pierre-Yves Strub
École Polytechnique
Paris, France
pierre-yves@strub.nu

Xiaodi Wu
University of Maryland
College Park, USA
xwu@cs.umd.edu

Li Zhou
MPI-SP
Bochum, Germany
zhou31416@gmail.com

ABSTRACT

EasyCrypt is a formal verification tool used extensively for formalizing concrete security proofs of cryptographic constructions. However, the EasyCrypt formal logics consider only classical attackers, which means that post-quantum security proofs cannot be formalized and machine-checked with this tool. In this paper we prove that a natural extension of the EasyCrypt core logics permits capturing a wide class of post-quantum cryptography proofs, settling a question raised by (Unruh, POPL 2019). Leveraging our positive result, we implement EasyPQC, an extension of EasyCrypt for post-quantum security proofs, and use EasyPQC to verify post-quantum security of three classic constructions: PRF-based MAC, Full Domain Hash and GPV08 identity-based encryption.

KEYWORDS

Formal verification, post-quantum cryptography

1 INTRODUCTION

The area of post-quantum cryptography (PQC) focuses on classical cryptosystems that are provably secure against quantum adversaries. PQC is based on computational problems that are conjectured to be hard for quantum computers, e.g., the learning with errors problem [38]. Simply relying on such assumptions, however, is insufficient to ensure security against quantum attackers; one must also verify that a security reduction holds in the quantum setting. The NIST competition and the challenges with PQC security proofs makes PQC a natural and timely target for computer-aided formal verification of such cryptosystems [3].

Several proof techniques used in the classical setting (for example, rewinding) are inapplicable in the quantum setting and so new approaches to prove security often need to be developed. Boneh et al. [15] and Song [42] characterized classes of classical security reductions that carry over to the quantum setting. However, even if the reductions to which their results apply is rather limited, it is not obvious whether the formal logics of verification tools, such as EasyCrypt, designed for the classical setting accurately capture the lifting of such proofs to the quantum setting. Perhaps more interestingly, an important class of proof techniques focusing on the quantum random-oracle model (QROM) [15] have been adapted from analogous classical ROM techniques [24, 36, 44, 53, 56]. The QROM is a variant of the random-oracle model [11] where adversaries can query the random oracle at many points simultaneously using quantum (superposition) queries.

A natural question to ask is therefore whether we need a fundamentally different approach to the design of formal verification tools to capture these results, which seem tantalizingly close to the classical setting. For example, Unruh [46] suggests that the EasyCrypt proof assistant [4, 5, 9], which has been used to mechanize security proofs for a broad set of constructions, is not sound for quantum adversaries. Concretely, [46] claim that the CHSH protocol, which is secure in the classic setting but not in the quantum setting, can be proved secure in EasyCrypt. While this point is moot, because the EasyCrypt logics was not designed (or claimed) to be sound for the quantum setting, it does raises an important question that we address in this paper.

Problem statement and contributions. Reflecting on [46], we ask:

(1) can we adapt the EasyCrypt program logic and libraries in a way that guarantees their soundness for PQC proofs?
(2) is the resulting framework expressive and practical to use?

Contrary to the suggestion of [46], we answer the first question in the affirmative by proposing post-quantum relational Hoare logic (pqRHL), a mild variant of EasyCrypt probabilistic relational Hoare logic (pRHL), and proving that pqRHL is sound for reasoning about quantum adversaries. By answering this question, we lay the foundations and develop tools for an approach that allows proving security of classic cryptographic constructions against quantum adversaries, while retaining the (relative) simplicity and benefits of existing tools.

We also answer the second question in the affirmative, by developing EasyPQC, an extension of EasyCrypt for post-quantum

CCS '21, November 15–19, 2021, Virtual Event, Republic of Korea
2021. ACM ISBN 978-1-4503-8454-4/21/11…$15.00
https://doi.org/10.1145/3460120.3484567

security proofs. Beyond pqRHL, EasyPQC relies on a small set of proof principles that are sufficiently powerful for carrying out PQC proofs and yet do not require reasoning about quantum programs — formal verification of quantum programs is still very challenging, especially for relational properties, and it is therefore essential for practicality to omit such details from the proofs whenever possible. Fortunately, we do not need to start from scratch. Indeed, a large part of the literature on QROM proofs of security implicitly follows this divide and proceeds in two steps. The first step establishes some general facts about an experiment with quantum adversaries, for instance bounding the probability of collisions being found by a quantum adversary. Proving such results requires some model and understanding of quantum computations. The second step performs the reduction proof using the results already established. This step hardly involves any reasoning about quantum computation at all; it only performs basic checks (e.g., that quantum oracles do not test their inputs—since in a quantum setting that would cause them to be modified). This two-step approach is supported by many existing techniques, including the semi-constant [54] and small-range [53]techniques, which can be supported by our approach.

Concretely, the main contributions of this paper are:

- A new implementation of EasyCrypt for verifying post-quantum security proofs. The cornerstone of the implementation is a new library that captures several common proof techniques in the QROM. In addition, the implementation includes several new functionalities, including support for sampling from a larger class of distributions and better support to reason about the complexity of constructed adversaries.
- A new interpretation of probabilistic relational Hoare logic that remains valid for post-quantum security proofs. Specifically, the interpretation of the logic remains valid when adversaries are interpreted as quantum programs. The key to the interpretation is a novel notion of coupling, called cq-coupling, which is used instead of the classic notion of probabilistic coupling to interpret judgments of the logic.
- Case studies which illustrate the workings of our approach. We provide mechanized proofs of PRF-based MAC [17] full domain hash signatures [12, 15] as well as the GPV08 identity-based encryption scheme [26, 54].

Overall, our work shows that the bulk of the infrastructure provided by EasyCrypt remains valid for quantum adversaries, and provides the first *practical* tool for computer-aided security proofs of PQC cryptosystems in the QROM.

Limitations and non-goals. We emphasize the goal of EasyPQC is to provide a practical tool for verifying many PQC proofs, but not achieving completeness, which adds more complexity to the system and yields cumbersome proofs. We give evidence that EasyPQC achieves its goal, in the sense that: 1. Many useful proof principles of PQC can be formalized as axioms; 2. Assuming these axioms, the program logic pqRHL can be used to verify many PQC security proofs.

Our current system is limited since we require the quantum queries to the QROM does not involve measurements. Nevertheless, it still captures some interesting constructions beyond the three mechanized examples. For instance in [1], the notion of a semi-classical oracle is introduced, where an event that determines the security bound is defined based on a measurement. We can capture such bounds in our system whenever they can be recast by moving the final measurement step in reduction to the adversaries and quantifying existentially over adversaries, as described in [13]. We provide more details in Section 8.

The EasyPQC implementation and examples can be downloaded at the following address: https://github.com/EasyCrypt/easycrypt/tree/deploy-quantum.

2 PRELIMINARIES

2.1 Basic Quantum Mechanics

Let $\mathcal{H}$ be a Hilbert space. In the finite-dimensional case, it is a complex linear space equipped with an inner product. Following the tradition in quantum computing, vectors in $\mathcal{H}$ are denoted in the Dirac form $|\psi\rangle$.

Any quantum system with finite degree of freedom is associated with a finite-dimensional Hilbert space $\mathcal{H}$ called its *state space*. When the dimensional is 2, we call such a system a *qubit*, the analog of bit in classical computing. A *pure state* of the system is described be a normalised vector in $\mathcal{H}$. When the state is not completely known but could be in one of some pure states $|\psi_i\rangle$ with respective probabilities p_i, we call $\{(p_i, |\psi_i\rangle)\}$ an *ensemble* of pure states or a *mixed state*, and the system is fully described by the *density operator* $\rho = \sum_i p_i |\psi_i\rangle\langle\psi_i|$. A quantum measurement $\mathcal{M}$ is described by a collection $\{M_i : i \in I\}$ of linear operators on $\mathcal{H}$, where I is the set of measurement outcomes.

2.2 QROM Background

Given two sets $\mathcal{X}$ and $\mathcal{Y}$, define $\mathcal{Y}^{\mathcal{X}}$ to be the set of functions $f : \mathcal{X} \to \mathcal{Y}$. Given a distribution D on a set $\mathcal{Y}$ and another set $\mathcal{X}$, define $D^{\mathcal{X}}$ as the distribution on $\mathcal{Y}^{\mathcal{X}}$, where the output for each input in $\mathcal{X}$ is chosen independently according to D.

The random-oracle model [11] is a widely used heuristic model for hash functions. In the classical setting, the adversary is given oracle access to a random function $O : \{0,1\}^* \to \{0,1\}^*$, and it can only learn a value $O(x)$ by querying the oracle O with input x. When the protocol is implemented in the real world, the random oracle is replaced by a concrete hash function. Therefore, in the post-quantum setting, it is reasonable to allow a quantum adversary to evaluate this hash function on quantum superpositions of inputs [15]. That is, we allow the adversary to submit a quantum state $|\psi\rangle = \sum \alpha_{x,y}|x, y\rangle$ to the oracle O and receive the state $|\psi\rangle = \sum \alpha_{x,y}|x, y \oplus O(x)\rangle$ in return. The superposition queries made by adversaries introduce obstacles in transforming classical ROM proofs to the QROM setting. For example, ROM techniques such as rewinding and extractability are not known to work straightforwardly in the quantum setting. Therefore, in the past decade, multiple new techniques have been developed for the QROM. In this work, we focus a particular set of these techniques:

- Small-range distributions [53]: rather than sampling $O(x)$ for every input x independently according to a distribution D, we can instead choose r elements $\{y_1, \ldots, y_r\}$ independently according to D, let $H : \mathcal{X} \to [r]$ be a random function, and then set $O(x) = r_{H(x)}$. Zhandry shows that when r is chosen appropriately based on the number of queries made by an adversary to O,

the adversary's interaction with O is indistinguishable in these two cases. See further details in Section 5.5.

- Semi-constant distributions [54]: For distribution D over range $\mathcal{Y}$, in semi-constant distribution, a fraction (indexed by $\gamma \in (0,1)$) of the domain $\mathcal{X}$ is set to a fixed element chosen uniformly from $\mathcal{Y}$, while the other elements of the domain are mapped to elements in $\mathcal{Y}$ according to distribution D. This semi-constant distribution is proven to be indistinguishable from a uniform function by algorithms making quantum oracle queries to the functions. We present the encoding details in Section 5.6 and give proof of the concrete bound we use in Appendix B.

While EasyPQC has the potential to mechanize other QROM techniques, such as one-way to hiding lemma [44] and compressed oracle [56], it is not our goal to capture the full extent of QROM in this work. We discuss some possible extensions of EasyPQC in Section 7.

3 MOTIVATING EXAMPLE

As a warm-up example we show how a security proof for the FDH signature scheme in the QROM [15] can be encoded in our system.

3.1 Example: Full-Domain Hash Signatures

Recall that a signature scheme consists of three probabilistic algorithms (keygen, sign, verify). The algorithm keygen generates a pair of public key pk and signing key sk. When given the signing key sk and a message m, the signing algorithm sign outputs a signature σ. The verification algorithm verify checks the validity of message/signature pair (m, σ) using the public key pk.

The definition of post-quantum unforgeability of a signature scheme in the QROM is described using a game, where a challenger generates a key pair and interacts with the adversary as follows. Upon receiving the public key pk, the adversary can issue classical queries m_i to the signing oracle and obtains (m_i, σ_i) as result. In the meanwhile, she can also issue quantum queries to the random oracle on a superposition of messages and obtains superposition of the hash values. In the end, the adversary needs to output a message/signature pair (m^*, σ^*), and we say the adversary wins the game if (m^*, σ^*) is a valid message/signature pair and m^* does not belong to the queries of the signing oracle. Without loss of generality, we assume that each message chosen by the adversary for the signing oracle is distinct.

The FDH signature scheme can be described as follows:

Definition 3.1 (FDH Signature). The FDH signature-scheme Σ is based on a permutation (kg, f, f^{-1}) and a hash function h. The algorithms (keygen, sign, verify) are defined in the following:

- keygen(): The key generation algorithm runs generation kg, to obtain the public key pk and the secret key sk.
- sign(sk, m): On input message $m \in \{0,1\}^*$, the signing algorithm outputs signature $\sigma = f^{-1}(\mathsf{sk}, h(m))$.
- verify(pk, σ, m): On input the public key pk and a message/signature pair (m, σ), the verification outputs 1 if $f(\mathsf{sk}, \sigma) = h(m)$ and 0 otherwise.

In this simpler version of the security proof in the QROM, the security of the FDH signature scheme assumes that the permutation f is a claw-free permutation. A claw-free permutation is given by a keygen algorithm and a pair of permutations $(f, f^{-1}), (g, g^{-1})$, where algorithm keygen produces a function key fk (the public key) and an inversion key ik (the secret key). f and g have the same domain, and the security of claw-free permutation is defined as: no QPT adversary can find a pair (x_0, x_1) such that $f(\mathsf{fk}, x_0) = g(\mathsf{fk}, x_1)$ given only fk. For brevity we do not give the formalization of this security game in EasyPQC.

In the classical setting, the unforgeability of the FDH signature scheme can be reduced to the security of the claw-free permuation in the random oracle model [21]. More specifically, if an adversary can break the signature scheme Σ, then there exists an attacker that can break the security of claw-free permutation.

Interestingly, the proof carries essentially without change to the post-quantum setting, assuming that the permutation is secure wrt to quantum adversaries. This is because the proof strategy does not imply modifying the distribution of random oracle outputs, but only to express the random oracle in a different, equivalent, form. We note that not all proof strategies that capture the same proof idea in the classical setting translate literally to the quantum setting; in particular, this proof can be presented in the classical world by lazy sampling and programming the random oracle on-the-fly, which cannot be done in the quantum world. However, the version we give here does not use this mechanism and, instead, it redefines the RO in one go, right at the beginning of the game, which can also be done in the quantum setting. We give the details next.

3.2 Formalization in EasyPQC

The formalization is presented in Figure 1, for conciseness we omit the declaration of local variables of procedures.

Basic formalization of the QROM. We start by explaining how the QROM is encoded. The types msg and hash represent the input type and output type of the random oracle. The module type QRO_T is the type of modules providing a procedure h. The keyword quantum denotes that the procedure can be called in superposition (it is implicit that the classical implementation of the oracle is lifted to the quantum setting in the expected way). Quantum adversaries running in the QROM will have access to an oracle of this type.

Then we provide a canonical implementation of a quantum random oracle (module QRO); this contains a variable fh, which is a function from msg to hash. Procedure h allows placing quantum queries to the pre-sampled hash function; on input x, it simply returns h x. Procedure init is used by security experiments for QROM initialization. The procedure initializes the variable fh by sampling a function following the distribution dfun dhash, where dhash is the uniform distribution over hash and dfun samples a function where, for each input, the output is independently sampled according to dhash. In other words, it is the distribution D^X defined in the previous section where D is dhash and X is msg.

Note that the oracle commits to the entire hash function at initialization, this is the major difference with classical ROM, where the oracle is generaly initialized lazily. This is the default implementation of the QROM, but during the security proofs we can use a modified version of it.

FDH and Existential Unforgeability. The module FDH provides an implementation of the FDH signature-scheme. Notice that the

```
module type QRO_T = { quantum proc h {_:msg} : hash }

op [lossless uniform full] dhash : hash distr.

module QRO : QRO_T = {
  var  fh : msg → hash
  proc init() = { fh ←$ dfun dhash; }
  quantum proc h {x:msg} = { return fh x; }
}.

module FDH (H:QRO_T) = {
  proc keygen() : pkey * skey = { k ←$ kg; return k; }
  proc sign(sk:skey, m:msg) : sign = { h ← H.h(m); return  f^-1 sk h;}
  proc verify(pk:pkey, m:msg, s:sign) : bool = {
    h ← H.h{m}; return h = f pk s; }
}.

module type OrclSign = { proc sign (m:msg) : sign }.

quantum module type AdvEUF (H:QRO_T) (S:OrclSign) = {
  proc main(pk:pkey) : msg * sign
}.

module EUF(A:AdvEUF) = {
  var log:msg list, m*:msg, pk:pkey, sk:skey;
  module S = FDH(QRO)
  module Os = {
    proc sign(m:msg) = { s ← S.sign(sk, m); log ← m :: log; return s; }
  }
  proc main() = {
    var s, b;
    QRO.init();  (pk, sk) ← S.keygen(); log ← [];
    (m*,s) ← A(QRO,Os).main(pk);
    b ← S.verify(pk, m, s);
    return b ∧ ¬m* ∈ log; }
}.
```

Figure 1: Security formalization for FDH

module is parametric in the implementation of the hash function (H), and that the different procedures do not have the keyword quantum, which means that only classical calls to those procedure are allowed.

The quantum module type AdvEUF describes the type of the quantum adversaries for the EUF game in the QROM. Such adversaries are parameterized by two oracles/modules H and S. Finally, the module capturing the unforgeability game EUF is defined; this implements the informal description provided before, intrumenting the sign oracle with a wrapper that logs the adversary's (classical) queries in variable log.

Security proof. The proof is structured as a sequence of games starting from EUF(A). In the following we assume that A can perform at most q_s queries to the sign oracle.[1]

The first transformation is a guess: we *modify* the initial game by sampling a function X that maps the message space to the Booleans; this function can be view as a subset of the message space. Here we use the distribution dfun (dbiased lam), where dbiased lam is the distribution over Booleans that return true with probability λ (the exact value of λ will be set later). The output of the game is then modified so that the adversary loses whenever one of two conditions occur: i. function X assigns false to the message used in the forgery or, ii. function X assigns true to any of the messages passed to the sign oracle. This leads us to game G, and a lemma where we have a multiplicative loss in the adversary's advantage that depends on the probability that the game declares the adversary a loser due to the modifications we introduced.

```
module G = {
  var X : msg → bool;
  proc main() = {
    b ← EUF(A, FDH).main(); X ←$ dfun (dbiased lam);
    return b ∧ X m* ∧ X ∩ log = ∅; }
}

lemma l1 : λ(1 − λ)^qs Pr[EUF(A) : res] ≤ Pr[G : res].
```

The proof of this lemma is done using the usual Hoare and union bound logic. The main difficulty is to show that the list log has size at most q_s, and dealing with distributions over functions; we have created a library to deal with the latter, and the resource analysis mechanism implemented in EasyCrypt is used to deal with the former. Notice that b implies that $m^* \notin$ log.

The next step allows to conclude the proof: we construct the following adversary for the claw free property. The adversary gets the evaluation key for the permutation and samples two random functions: X as above and and a function hs that maps messages to the inputs of the permutation.

The definition of variable fh (used by QRO) depends on f, g, the evaluation key and hs. Intuitively, the oracle uses hs to sample a preimage m for the permutation and uses function X to decide whether to produce the final hash value using g (with probability λ) or f with probability $(1 - \lambda)$. Note that, for all messages where the random oracle outputs f pk (hs m), the correct signature value is hs m.

```
module B (A:AdvEUF) = {
  var hs : msg → sign, bf : msg → bool

  module Os = { proc sign(m:msg) = { log ← m :: log; return hs m; } }

  proc main(pk:pkey) = {
    var s
    X ←$ dfun (dbiased lam); hs ←$ dfsign;  log ← [];
    QRO.fh ← (fun m ⇒  if X m then g pk (hs m) else f pk (hs m));
    (ms,s) ← A(QRO, Os).main(pk);
    return (s, hs ms);
  }
}.

equiv G_CFB : G ~ ClawFree(B(A)) : ={glob A} ⇛  res{1} ⇒  res{2}.

lemma l2 : Pr[G : res] ≤ Pr[ClawFree(B(A)) : res].
proof. byequiv G_OW. qed.
```

The proof uses our new pqRHL that can handle quantum adversaries. The logic is relational, which means that we can reason about the relation between the executions of two programs. The precondition of the judgment is ={glob A}, which means that the global state of the adversary should be initially the same. Since the adversary is quantum, this equality over a quantum state. The post-condition of the judgment relates the final states: it states that, whenever the result of the first game (G) is true, the result of the second game is also true.

The proof of the relational judgment in EasyPQC take about 20 lines. The intuition is the following: we should show that the view of the adversary A is identical in both games. Its view is fixed by its inputs and the implementation of its oracles. The inputs are easily shown to be identically distributed, so the more intricate part of the

[1] We omit the details on how to ensure this in EasyPQC.

analysis implies showing that the outputs of the oracles are also identically distributed.

For the hash oracle, game G samples the function h uniformly, while B uses the procedure described above; however, since f and g are two bijective functions, the distribution the hash function is uniform in both games. The same reasoning can be done for the sign oracle, but we must use the fact that we have excluded the possibility that signature queries include messages that are not in X. To see this, note that the sign oracle in B returns hs m regardless of whether the message is in X or not. However, in game G, the result is $(f^{-1}\ \mathsf{sk}\ (\mathsf{fh}\ \mathsf{m}))$, which is always guaranteed to be a valid signature. One might be tempted to conclude that the distributions in both games are the same (f^{-1} is bijective), but this is not correct. The problem is that it is not sufficient to show that the distribution of the hash oracle and the distribution of the sign oracle are pairwise identical (in isolation). We must prove that the joint distribution is identical, as this is the adversary's view. We can only prove this once we use the modification to the game made in the previous step, which guarantees that A will lose in G if one request made to the sign oracle is in X. In other words the view of the adversary A changes only when game G declares it a loser anyway.

The pRHL judgment G_CFB allows to relate the winning probability of the two games (lemma l2). By combining both results we derive the following conclusion:

lemma conclusion : **Pr**[EUF(A) : **res**] $\leq \frac{\Pr[\mathsf{ClawFree}(\mathsf{B}(\mathsf{A})) : \mathsf{res}]}{\lambda(1-\lambda)^{q_S}}$

The result is true for all $\lambda \in [0, 1]$ but the bound is minimal for $\lambda = \frac{1}{q_s+1}$. Notice that the reduction is not efficient because the adversary B samples two functions (one from message to signature and the other from message to bool). This is because the message space is large, which makes oracles being chosen exponential-sized objects. In Section 5.2 we discuss how to prove that this reduction can be implemented efficiently.

4 POST-QUANTUM RELATIONAL HOARE LOGIC

This section presents pqRHL, a new relational program logic for reasoning about QROM and quantum adversaries. From a user perspective, our logic is very closely related to the pRHL already implemented on EasyCrypt, and there are few noticeable differences. For instance, one cannot do a case analysis on the adversary state; Moreover, one must check that assertions satisfy a technical condition (CM, closed under mixtures), that we will explain shortly. However, proving the soundness of the relational logic is substantially more complex. The main reason for this complexity is the adversary rule, which must be proved sound for all valid instances of adversaries. This is done by induction on the structure of valid adversaries, using the soundness of the rules for other program constructs. In a post-quantum setting, valid adversaries are quantum programs; therefore, the relational program logic should be able to reason about quantum programs. Unfortunately, relational logics for quantum programs are complex, and rather different from the pRHL implemented by EasyCrypt. For instance, their assertions may be modelled as (quantum) expectations rather than plain boolean-valued predicates. In this work, we focus on post-quantum cryptography in the QROM and consider nonspecified adversaries (quantifying over all possible adversaries), so equality is sufficient for most of the cases. This allows us to build a logic that retains the flavour and automation of pRHL. However, a key hurdle remains: we must prove that the logic is sound. This, in particular, needs to define a well-behaved notion of validity in the spirit of the coupling-based notion of validity used for pRHL. Our proposal is the new notion of cq-coupling, which achieves the desired aim. On the other hand, relational logic also has been used for reasoning about more general quantum programs beyond the target of this work, where quantum equality is not expressive enough. We point out that our core logic has the potential to be extended for formalizing these programs.

4.1 Programming Language

The semantics of cq statements has been studied in [25] based on the concept of classical-quantum state (cq-state). Briefly, cq-state is the mapping from classical evaluations to partial density operators (the mathematical object that describes both pure/mixed quantum states), which is a direct extension of probability distribution in the probabilistic setting. We review the semantics in the appendix.

We point out that the cq statements are universal for quantum computation, so any valid quantum adversary can be described by the syntax. On the other side, we restrict that the only allowed quantum commands except in adversary call is $\overline{q} \mathrel{*}= \mathrm{Uc}[f]$ which is sufficient to model QROM; this avoids complex matrix calculations which is inevitable in reasoning about general quantum initialization, unitary transformation and quantum measurement.

We follow the usual paradigm of classical control and quantum data, and consider a core programming language with classical control flow structures (loops and conditionals), calls (classic and quantum oracles, adversaries), and atomic operations (assignments, sampling, initialization, unitary transformations and measurement). The syntax of classic-quantum commands is as follows:

Definition 4.1 (Classical-Quantum(cq for short) statements, c.f. [25]). The set **Cmd** of classical-quantum statements is defined by the following syntax:

$c ::=$	skip	no op
	$x := e$	deterministic assignment
	$x \xleftarrow{\$} d$	probabilistic assignment
	$q := e$	quantum initialization
	$\overline{q} \mathrel{*}= U$	unitary transformation
	$x := \mathsf{meas}\ M[\overline{q}]$	quantum measurement
	$c; c$	sequencing
	$\mathsf{if}\ e\ \mathsf{then}\ c\ \mathsf{else}\ c$	conditional
	$\mathsf{while}\ e\ \mathsf{do}\ c$	while loop
	$x \leftarrow \mathcal{A}(e)$	adversary call
	$x \leftarrow \mathcal{Q}(e, \overline{q})$	oracle/procedure call
	$\mathsf{return}\ e$	classic return expression

The quantum commands above obey that quantum variables/registers only appear in at most one side of the commands, which enables us avoid using a linear type checker to exclude physically unrealistic commands; for example, quantum assignment $q_1 := q_2$ that duplicate quantum state of q_2 to q_1 is illegal since it violates the no-cloning theorem. We explain some of the constructs in more detail:

- Quantum initialization: this statement first evaluates the classical expression e, then initializes the quantum variable q to $|e\rangle$.
- Unitary transformation: this statement applies unitary transformation U on register $\overline{q}$. We allow U to be parameterized by classical variables. In order to model QROM, we introduce a simple notation $\overline{q} *= \mathrm{Uc}[f]$, which is a unitary transformation that maps $|x, y\rangle$ to $|x, y \oplus f(x)\rangle$.
- Quantum measurement: this command performs the quantum measurement M on register $\overline{q}$ and assigns the (classical) output to x. Note that measurement not only extracts classical information from a quantum state but also modifies the quantum state.
- Adversary calls: we require that both input and output are classical but the adversary may perform quantum computations. Different from oracle/procedure call, adversary is a non-specified program.
- Oracle/procedure call: $x \leftarrow Q(e, \overline{q})$ takes a classical expression e and a possible quantum register $\overline{q}$ as input and assigns the returned classical value to x. Since the state on the quantum register is no-cloneable, $\overline{q}$ is actually a reference parameter rather than actual parameter; it is a pointer to the corresponding quantum systems. For simplicity, if there is no classical input and output, we just write it as $Q(\overline{q})$.

For the limitation of this constraint, see Sec. 7 for more discussion.

4.2 Assertions and Judgment

Relational assertions are first-order formulae over classical and quantum predicates. However, the **only** predicate on quantum states is global equality (we can easily extend it to more general cases, e.g., using projections as the predicate on quantum states, but here we only focus on equality).

Definition 4.2 (Syntax of relational assertion). Relational assertion Φ are first-order logic over following Atomic Formulas (AF) with all bound variables being classical.

$$\begin{aligned} \mathrm{AF} \equiv \mid\ & e\langle 1/2\rangle = e\langle 1/2\rangle \mid e\langle 1/2\rangle < e\langle 1/2\rangle \mid e\langle 1/2\rangle \in e\langle 1/2\rangle && \text{(in)equality of classical expressions} \\ \mid\ & O(e\langle 1/2\rangle, \cdots, e\langle 1/2\rangle) && \text{other classical predicates} \\ \mid\ & (=_Q) && \text{global quantum equality} \end{aligned}$$

The classical free variables free (Φ) is defined as usual. We write $\mathrm{cls}(\Phi)$ if Φ does not contain quantum equality.

One challenge when designing the logic was to avoid imposing any syntactic restriction on logical formulas. Although restrictions on the use of quantum assertions would considerably ease the proof of soundness of the logic, they would have a negative effect on the usability of the logic. For instance, they would complicate the use of a weakest precondition calculus, since one would need to ensure that every intermediate assertion computed by the calculus satisfies the restrictions. The solution presented here allows quantum equality to be compounded with any other formulas through logical connectives, thereby avoiding any problem. This stands in contrast with [10, 46] where assertions are projections/observables and are not closed under logical implication.

We now turn to the logic. Judgments are of the form

$$\models c_1 \sim c_2 : \Phi \Rightarrow \Psi$$

where c_1 and c_2 are cq statements, Φ and Ψ are relational assertions.

Validity of judgments is based on the concept of coupling [32, 34, 43, 48, 49, 58], which is employed to interpret judgments of relational logic in both probabilistic [6, 8] and quantum [10, 33, 46, 47] setting. We define cq-coupling as a combination of both probabilistic and quantum couplings. Briefly, a product cq-state $\Delta_\times$ is a cq-coupling of two cq-states (Δ_1, Δ_2) if its marginals are Δ_1 and Δ_2; see Appendix for details. In the remainder, we use the notation $\Delta_1 [\![\Phi]\!]^\# \Delta_2$ to state the existence of a cq-coupling of (Δ_1, Δ_2) that satisfies Φ and $[\![\Phi]\!]$ for the interpretation of the assertion Φ.

Definition 4.3 (Validity of Judgment). Judgment $\models c_1 \sim c_2 : \Phi \Rightarrow \Psi$ is valid, if for any two classical-quantum memories[2] Δ_1 for c_1 and Δ_2 for c_2 such that $(\Delta_1, \Delta_2) \in [\![\Phi]\!]$, the output pairs $[\![c_1]\!](\Delta_1)$ and $[\![c_2]\!](\Delta_2)$ are related by cq-lifting $[\![c_1]\!](\Delta_1)[\![\Psi]\!]^\#[\![c_2]\!](\Delta_2)$.

A valid judgment gives a way to derive probabilistic (in)equality. Any classical relational assertion Φ, i.e., Φ contains no quantum equality ($=_Q$), is fully determined by classical memories **CMem**, thus it can also be interpreted as a relation $[\![\Phi]\!]_c \subseteq \mathbf{CMem} \times \mathbf{CMem}$. Recall the fundamental lemma of cq-lifting with classical relational assertion:

Lemma 4.4. *Let $E_1, E_2 \subseteq \mathbf{CMem}$. Let Φ be a classical relational assertion such that for every $(\sigma_1, \sigma_2) \in \mathbf{CMem} \times \mathbf{CMem}$, $(\sigma_1, \sigma_2) \in [\![\Phi]\!]_c$ implies $\sigma_1 \in E_1 \implies \sigma_2 \in E_2$. If $\Delta_1 [\![\Phi]\!]^\# \Delta_2$ then $\Pr_{\Delta_1}[E_1] \leq \Pr_{\Delta_2}[E_2]$.*

Based on it, we are able to bound probabilities of events via certain post-conditions. For example, the following lemma establishes a probabilistic inequality between two events in two games and is useful in reduction steps. See Appendix for more lemmas.

Lemma 4.5. *If $\models c_1 \sim c_2 : \Phi \Rightarrow \phi_1\langle 1\rangle \implies \phi_2\langle 2\rangle$ and ϕ_1 and ϕ_2 are classical, then for every two classical-quantum memories Δ_1 and Δ_2 such that $(\Delta_1, \Delta_2) \in [\![\Phi]\!]$, we have: $\Pr_{[\![c_1]\!](\Delta_1)}[[\![\phi_1]\!]] \leq \Pr_{[\![c_2]\!](\Delta_2)}[[\![\phi_2]\!]]$.*

4.3 Inference Rules and Soundness

Our setting is carefully crafted to make the proof system of our core logic very similar to the pRHL logic implemented by EasyCrypt, with extra and different rules listed in Figure 2 for quantum constructs and Figure 3 for reasoning about adversaries.

Construct and Structural Rules, Figure 2. [QROM] is used for reasoning about same quantum unitary transformations and [QROM-L] states that classical assertions are preserved under unitary transformation. For a procedure call $x \leftarrow Q(e, \overline{q})$, $\mathbf{args}_Q$ and Res_Q are the classical input argument and classical returned expression of Q; and $\mathbf{body}_{Q(\overline{q})}$ is the body of Q with reference parameter (quantum register) $\overline{q}$. In [Case], e is an event and thus a classical assertion, where event typically refers to the classical event since physically we cannot look at the quantum state directly. [PTrans] only works for classical assertions since the composition of assertions containing quantum equality are generally undefined. There are two frame rules [PFrame] and [QFrame] for dealing with the cases that invariant Θ is classical (denoted by $\mathrm{cls}(\Theta)$) and the programs are probabilistic (denoted by $\mathrm{prob}(c_1)$ and $\mathrm{prob}(c_2)$, i.e., c_1 and c_2

[2] We remark that its classical memory of input is not probabilistic, but we allow possible ensembles of quantum memory since we cannot look at quantum state directly.

consist of classical statements and probabilistic assignment but lack of any quantum statements); $\mathrm{MV}(c)$ is the set of possibly modified variables of c .

Rules for adversaries, Figure 3. [QAdv] is used for reasoning about adversary with "equivalent" procedure call, i.e., using the same classical/quantum oracles. [QAdv-Bad] provides a way for up-to-bad reasoning, i.e., the oracles adversary calls is "equivalent" up to some bad event β, then we can gather β to the post-condition. Different from its analog in pRHL, the non-specified adversary might perform quantum computations and query a quantum-access oracle via procedure call; thus $\mathrm{eqmem}_{\mathcal{A}}$ asserts that both the classical and quantum memories are equal. We require that the invariant Ψ and bad event β are classical since the input and output of the adversary are both classical. Further, we write $\mathrm{ast}(\mathbf{body}_{\mathcal{A}})$ to ensure that the body of adversary $\mathcal{A}$ is lossless, i.e., it should almost surely terminate. $\mathrm{LMV}(\mathcal{A})$ is the set of possibly "locally" modified variables of $\mathcal{A}$, i.e., $\mathcal{A}$ cannot change any variables out of $\mathrm{LMV}(\mathcal{A})$ except making procedure calls; thus side-condition $\mathrm{LMV}(\mathcal{A}) \cap (\mathrm{free}(\Psi) \cup \mathrm{free}(\beta)) = \emptyset$ ensures that local computation of $\mathcal{A}$ will not affect Ψ and β.

Comparison to pRHL: It worth pointing out that the rest of the rules are the same as in pRHL, including construct rules for classical/probabilistic statements [Skip], [Seq], [Assn], [Rand], [Cond], [While] and their one-side rules, and structural rules [False], [True], [Conseq], [Exists], [Disj]. Therefore, much of EasyCrypt can be reused to implement our core logic.

Though the inference rules we present are quite similar to pRHL, proving the soundness is quite different since we face a great challenge of dealing with closed-under-mixture condition which is defined as follows:

Definition 4.6. A formula Φ is closed under mixtures (CM), written $\mathrm{cm}(\Phi)$, if for a finite or countably infinite sequence of classical-quantum states Δ_i such that $\Delta_i \models \Phi$ and $\lambda_i \in [0,1]$ such that $\sum_i \lambda_i = 1$, then their mixture $\sum_i \lambda_i \Delta_i \models \Phi$.

Since the execution path of the probability/quantum program is non-deterministic, the final output state after the conditional is a mixture of every output from each branch, and thus we generally need an extra side-condition $\mathrm{cm}(\Psi)$ to ensure that the mixture of outputs also satisfies Ψ. In probabilistic setting, every assertion is CM, but this is not the case in quantum setting. To avoid checking CM in relational proofs, we prove the soundness theorem with an additionally requirement that the final post-condition is CM while reusing the existing proof systems of pRHL without any CM conditions:

Theorem 4.7 (Soundness). $\vdash c_1 \sim c_2 : \Phi \Rightarrow \Psi$ *and* $\mathrm{cm}(\Psi)$ *implies* $\models c_1 \sim c_2 : \Phi \Rightarrow \Psi$.

This ensures that we can first use the proof system to derive the formulas without caring about whether the intermediate assertion is CM, and finally we just check whether the post-condition is CM. In fact, in post-quantum cryptographic verification, what we ultimately care about is the probability of classical events which can be described by a formula that does not contain quantum predicates and thus it is always CM.

$$\frac{\Psi \implies f\langle 1\rangle = f\langle 2\rangle}{\vdash \overline{q} \mathrel{*}= \mathrm{Uc}[f] \sim \overline{q} \mathrel{*}= \mathrm{Uc}[f] : (=_Q) \wedge \Psi \Rightarrow (=_Q) \wedge \Psi} \text{ [QROM]}$$

$$\frac{\mathrm{cls}(\Psi)}{\vdash \overline{q}_1 \mathrel{*}= \mathrm{Uc}[f] \sim \mathrm{skip} : \Psi \Rightarrow \Psi} \text{ [QROM-L]}$$

$$\frac{\begin{array}{l}\Phi \implies \Phi'[e_1\langle 1\rangle/\mathbf{args}_{Q_1}\langle 1\rangle][e_2\langle 2\rangle/\mathbf{args}_{Q_2}\langle 2\rangle] \\ \vdash \mathbf{body}_{Q_1(\overline{q}_1)} \sim \mathbf{body}_{Q_2(\overline{q}_2)} : \Phi' \Rightarrow \\ \qquad \Psi[\mathbf{res}_{Q_1}\langle 1\rangle/x_1\langle 1\rangle][\mathbf{res}_{Q_2}\langle 2\rangle/x_2\langle 2\rangle]\end{array}}{\vdash x_1 \leftarrow Q(e_1, \overline{q}_1) \sim x_2 \leftarrow Q(e_2, \overline{q}_2) : \Phi \Rightarrow \Psi} \text{ [QCall]}$$

$$\frac{\begin{array}{l}\Phi \implies \Phi'[e_1\langle 1\rangle/\mathbf{args}_{Q_1}\langle 1\rangle] \\ \vdash \mathbf{body}_{Q_1(\overline{q}_1)} \sim \mathrm{skip} : \Phi' \Rightarrow \Psi[\mathbf{res}_{Q_1}\langle 1\rangle/x_1\langle 1\rangle]\end{array}}{\vdash x_1 \leftarrow Q(e_1, \overline{q}_1) \sim \mathrm{skip} : \Phi \Rightarrow \Psi} \text{ [QCall-L]}$$

$$\frac{\vdash c_1 \sim c_2 : \Phi \wedge e \Rightarrow \Psi \qquad \vdash c_1 \sim c_2 : \Phi \wedge \neg e \Rightarrow \Psi}{\vdash c_1 \sim c_2 : \Phi \Rightarrow \Psi} \text{ [Case]}$$

$$\frac{\begin{array}{l}\vdash c_1 \sim c : \Phi_1 \Rightarrow \Psi_1 \qquad \vdash c \sim c_2 : \Phi_2 \Rightarrow \Psi_2 \\ \mathrm{cls}(\Phi_1) \quad \mathrm{cls}(\Phi_2) \quad \mathrm{cls}(\Psi_1) \quad \mathrm{cls}(\Psi_2)\end{array}}{\vdash c_1 \sim c_2 : \Phi_1 \circ \Phi_2 \Rightarrow \Psi_1 \circ \Psi_2} \text{ [PTrans]}$$

$$\frac{\vdash c_1 \sim c_2 : \Phi \Rightarrow \Psi \qquad \mathrm{cls}(\Theta) \qquad \mathrm{free}(\Theta) \cap (\mathrm{MV}(c_1)\langle 1\rangle \cup \mathrm{MV}(c_2)\langle 2\rangle) = \emptyset}{\vdash c_1 \sim c_2 : \Phi \wedge \Theta \Rightarrow \Psi \wedge \Theta} \text{ [PFrame]}$$

$$\frac{\begin{array}{l}\vdash c_1 \sim c_2 : \Phi \Rightarrow \Psi \qquad \mathrm{prob}(c_1) \qquad \mathrm{prob}(c_2) \\ \mathrm{free}(\Theta) \cap (\mathrm{MV}(c_1)\langle 1\rangle \cup \mathrm{MV}(c_2)\langle 2\rangle) = \emptyset\end{array}}{\vdash c_1 \sim c_2 : \Phi \wedge \Theta \Rightarrow \Psi \wedge \Theta} \text{ [QFrame]}$$

Figure 2: Selected rules for cq-commands.

5 IMPLEMENTATION

5.1 Extensions to EasyCrypt

In this section we describe the main modifications to EasyCrypt. Naturally we have extended the syntax of EasyCrypt to allow the the declaration of modules that represent quantum adversaries and the declaration of quantum procedures (procedures specified using a classical implementation, but which can be called in superposition by quantum modules). Our choice of keeping the lifting of classical oracles to the quantum setting implicit in the code has the signicant advantage that we do not need to extend the programming languages in EasyCrypt to allow expressing the details of quantum computations. We now give a short summary of how the implementation in EasyCrypt matches the formal logics.

Quantum procedures. We now allow declaring procedure types as being quantum. These procedures can take two kinds of arguments: classical arguments taken in parenthesis and quantum arguments taken in curly brackets. Local variables of a procedure can also be declared as quantum.

```
var count : int

quantum proc f(i:int){x:t} : u = {
  quantum var y : u
  count ← count + i
```

$$\frac{\forall Q, y, z, \overline{q}.\ \vdash y \leftarrow Q(z,\overline{q}) \sim y \leftarrow Q(z,\overline{q}) : (=_Q) \wedge z\langle 1\rangle = z\langle 2\rangle \wedge \Psi \Rightarrow (=_Q) \wedge y\langle 1\rangle = y\langle 2\rangle \wedge \Psi \quad \Theta \triangleq \Psi \wedge \mathsf{eqmem}_{\mathcal{A}} \quad \mathsf{cls}(\Psi) \quad \mathrm{LMV}(\mathcal{A}) \cap \mathrm{free}(\Psi) = \emptyset}{\vdash x_1 \leftarrow \mathcal{A}(e_1) \sim x_2 \leftarrow \mathcal{A}(e_2) : e_1\langle 1\rangle = e_2\langle 2\rangle \wedge \Theta \Rightarrow x\langle 1\rangle = x\langle 2\rangle \wedge \Theta}\ [\textsc{QAdv}]$$

$$\frac{\begin{array}{c}\forall Q, y, z, \overline{q}.\ \vdash y \leftarrow Q(z,\overline{q}) \sim y \leftarrow Q(z,\overline{q}) : (=_Q) \wedge z\langle 1\rangle = z\langle 2\rangle \wedge \Psi \wedge \neg\beta \Rightarrow ((=_Q) \wedge y\langle 1\rangle = y\langle 2\rangle \wedge \Psi) \vee \beta \\ \vdash y \leftarrow Q(z,\overline{q}) \sim \mathsf{skip} : \beta \Rightarrow \beta \qquad \vdash \mathsf{skip} \sim \overline{q} \leftarrow Q(z,\overline{q}) : \beta \Rightarrow \beta \\ \Theta \triangleq \Psi \wedge \mathsf{eqmem}_{\mathcal{A}} \quad \mathsf{cls}(\Psi) \quad \mathsf{cls}(\beta) \quad \mathrm{ast}\left(\mathbf{body}_{\mathcal{A}}\right) \quad \mathrm{LMV}(\mathcal{A}) \cap (\mathrm{free}(\Psi) \cup \mathrm{free}(\beta)) = \emptyset\end{array}}{\vdash x_1 \leftarrow \mathcal{A}(e_1) \sim x_2 \leftarrow \mathcal{A}(e_2) : e_1\langle 1\rangle = e_2\langle 2\rangle \wedge \Theta \Rightarrow (x\langle 1\rangle = x\langle 2\rangle \wedge \Theta) \vee \beta}\ [\textsc{QAdv-Bad}]$$

Figure 3: Derived rules for reasoning about adversary.

```
  if (i = 0) y ← c;
  else y ← H.h{x};
  return y;
}
```

Here i is a classical variable whereas x and y are *quantum*. The type system of EasyPQC ensures that the control flow does not depend on quantum variables; it also ensures that assignments performed to classical variables do not depend on quantum variables. For example, the assignment performed to count cannot depend on q and the condition of the if statement cannot depend on q. Naturally a quantum variable can receive the result of a quantum procedure.

One can notice that the language used here is not the same than the one used in the formal logics introduced in the previous section. However, the typing restrictions ensure that any quantum procedure described in the syntax of EasyPQC can be translated into that formalism. The idea is to lift each quantum local variable to a function. For example, the procedure f can be translated as follows:

```
f(i, |x,y>) = {
  var fy : t → u;
  count ← count + i;
  if (i = 0) fy ← (fun x' ⇒ c);
  else fy ← H.h(); // this now returns a functional operator
  |x,y> *= Uc[fy]
}
```

Naturally, using such quantum values to determine control flow or in an assignment to a classical variable is not allowed. However, the EasyCrypt language can be used to describe simple transformations over these states: intuitively they correspond to using classical code to sample a function fy from a distribution over functions, and then using that function to transform the quantum state using the standard unitary transformation Uc[fy].

Changes in the core rules of the logic. The implementation of EasyCrypt is based on a set of core rules; all the remaining complex/automating reasoning is checked wrt to those rules. So just this core logic needs to be modified carefully. The formal logics we give in this paper were designed so that the sub-part of the rules that need to be implemented in EasyCrypt is very close to the original ones. Indeed, the modifications to the core logics of EasyCrypt consisted in adding extra side conditions that are specific to the quantum setting, which restrict the application of the rules.

An example of these side conditions is closure under mixture required by the soundness theorem 4.7. The soundness theorem is used each time that we deduce a fact on probability from a pRHL judgment. Our implementation checks the closure under mixture predicate using an (incomplete) syntactic check. Some other rules need to be patched, like the [Case] which imposes that the condition is classical. Finally, it was necessary to modify some rules of the other logics provided by EasyCrypt. In particular we restrict the union bound rule to classical oracles; this rule allows to bound the probability of an event that can be triggered in an oracle using a bound on the number of calls.

Distributions over functions. As seen above, reasoning in the post-quantum setting implies reasoning about distributions over functions intensively. For this reason we have added a new library to the core theories of EasyCrypt; this library formalizes how to sample a function following the distribution D^X in term of simple constructions (the type X needs to be finite). The theory includes a set of basic lemmas, which allows reasoning about distributions over functions essentially in the same way as we reason about other distributions in EasyCrypt.

5.2 A library for QROM reasoning

Our machine-checked security proofs rely on a formalization of the QROM execution model and four fundamental results that apply to all quantum algorithms executing in this model. In this section we give a brief overview of this QROM library and report on our efforts to reduce the number of axioms that we introduce.

The QROM execution model is formalized by providing all algorithms with access to an oracle of type QRO; for readability we recall the definition here and refer the reader back to Section 3.2 for the explanation.

We take some liberty with the syntax of EasyPQC for readability.

```
type from.
type hash.

op [lossless uniform full] dhash : hash distr.

module type QRO_i = {
  proc init () : unit
  quantum proc h {_:from} : hash
}.

module type QRO_T = { include QRO_i [h] }.

module QRO : QRO_i = {
  var  fh : from → hash

  proc init() = { fh ←$ dfun fhash; }

  quantum proc h {x:from} = { return fh x; }
}.
```

In what follows we describe four EasyCrypt theories that allow us to carry out game hops that profoundly modify the operation of the QROM.

5.3 Fundamental lemma and efficient sampling

One of the crucial aspects of proofs in the QROM is that reductions must be able to efficiently simulate a (possibly modified) random oracle. We provide a formalization of the general result in [55, Theorem 4.5] that permits carrying out a conservative game hop (i.e., a hop without any loss) from a game that samples a random function initially, to another one where the game computes the answers to the adversary's queries to that function on the fly.

We prove in EasyCrypt that this lemma can be reduced to the following fundamental Theorem given in [55, Lemma 4.1]. Here, QRO_main_D is an experiment that takes an arbitrary distribution (adfhash)for sampling the hash function used in the QROM module, runs a quantum adversary that interacts with the oracle, and outputs whatever the adversary returns on termination. Note the type of the adversary, which states the upper bound q on the number of QROM queries, and the notation AdvRO{-QRO} indicate that we quantify over all adversary of type AdvRO that do not have a direct access to the global variable used by the module QRO.

```
module type AdvRO (H:QRO_T) = { proc main () : result { H.h^q} }

axiom dA_split (A<:AdvRO{-QRO}):
  ∃ (C : (from * hash) list → result → real),
  ∀ (adfhash : (from → hash) distr) (r:result),
    Pr[ QRO_main_D(A).main(adfhash) : res = r] =
      ∑{l | size l=2q} C l r * Pr[f ← adfhash : ∀(x, r) ∈ l, f x = r].
```

The efficient simulation theorem then states that any quantum adversary interacting with a QROM behaves in the same way if it is given access to the concrete oracle below.

```
module LQRO : QRO_i = {

  proc init() = { seed ←$ genseed; QRO.fh ← (fun x ⇒ compute seed x); }

  include QRO [h]
}.

lemma efficient_sim (A<:AdvRO{-QRO}) (r : result):
  Pr[ IndQRO(A,QRO) : res = r ] = Pr[ IndQRO(A,LQRO) : res = r ].
```

Here, compute is an operator that generates a row in a Vandermonde matrix with $2q$ columns based on the input x, and computes the inner product with an initially sampled random seed. This theory assumes that the hash input and hash output are the same finite field, over which the Vandermonde matrix is defined. However, based on this result it is easy to derive corollaries for any hash input that can be injectively mapped to the finite field, as well as distributions over hash output types that can be efficiently generated from the uniform distribution over the finite field. For example, one can sample a biased Boolean value by comparing the finite field value with respect to an appropriate threshold.

The seed is a vector of length $2q$ over the same finite field. The proof relies on the fact that any $2q$ rows in such a matrix are linearly independent. We state this as an additional axiom that expresses this property as invertibility/bijection of the hash computation.

```
op uhash_list(xs : ff_in list, seed : vector) : vector =
  offunv (fun i ⇒
        let x = nth witness xs i in
        let xv = offunv (fun (j : int) ⇒  x^j) in dotp xv seed).

axiom indep xs:
uniq s ⇒  size s = size vs ⇒  size s < p ⇒
  uhash_list_inv s (uhash_list s vs) = vs ∧
  uhash_list s (uhash_list_inv s vs) = vs.
```

5.4 Collision Finding

We define a collision finding game for an adversary interacting with the random oracle in the natural way, and express the hardness of finding collisions as the axiom below, which corresponds to [55, Theorem 4.9]. We allow the adversary an auxilliary input, which is typically information required to construct a reduction to this property. In the following, r is the probability to obtain a element using the distribution dhash (the distribution is assumed to be uniform).

```
quantum module type AdvCol (H:QRO) = {
  proc main(i:input) : from * from {H.h^q}
}.

module Col(A:AdvCol) = {
  proc main(i:input) = {
    var xx;
    QRO.init();
    (x1, x2) ← A(QRO).main(i);
    return (x1 ≠ x2 ∧ QRO.fh x1 = QRO.fh x2);
  }
}.

axiom pr_col (A<:AdvCol{-QRO}): Pr[Col(A) : res] ≤ 27*(q+2)^3 * r.
```

Note that the classical version of this lemma, which is known as a switching lemma, can be proved in EasyCrypt by applying a tactic that allows to upper-bound the probability of an event in a sequence of oracle calls by proving a bound on the probability of this event in a single call. However, such a rule does not exist for quantum adversaries. Indeed, for a quantum procedure, we do not have a way to express the probability of an event occurring in a quantum procedure call.

5.5 Small-Range Distributions

We formalize two versions of the small-range distribution technique for modifying an oracle which samples output hash values according to some distribution dhash. Recall that the idea is to fix an integer r, and to modify the random oracle to sample hash values in two steps: one first samples a value in the range $[r] = 0..r-1$, and then use a random function based on dhash to define outputs for each value in $[r]$. The first version corresponds to [55, Corollary 4.15]. Below we give the two oracles with which the quantum adversary may interact and the distinguishing bound of any quantum adversary.

```
op difun r = dfun [0..r-1].

quantum module type AdvSR (H:QRO_T) = {
  proc main(r:int) : bool { H.h^q }
}.

module SR : SR_i = {
  var rh : hash list
  var fr : from → int
  proc init (r:int) = {
    rh ←$ dlist dhash r;
```

```
  fr <$ difun r;
  QRO.h <- fun x => nth witness rh (fr x);
 }
}.

module SRO : SR_i = {
 proc init (r:int) = {
  SR.fr <$ difun r;
  QRO.init();
 }
}.

axiom advantage q r (A<:AdvSR{-SR, -QRO}):
 0 < r =>
 | Pr[IND_SR(SRO,A).main(r) : res] - Pr[IND_SR(SR,A).main(r) : res] | <= 27 * q^3 / r.
```

Here, dlist takes a distribution and creates a distribution over lists of the given length (in this case r), which samples independently each element in the list from the input distribution. The full oracle is therefore samping one such list and then picking one element at random for each input.

The second version of the assumption corresponds to the strengthening in [55, Theorem 4.16], where the adversary receives also (fr x) as an answer to its queries (this means that fr must be sampled, even in the oracle where it is not actually used to compute the output hash values). The bound we use for this variant was inferred from the proof given in [55].

```
quantum module type AdvSRr (H:SRr) = {
 proc main(r:int) : bool { H.hq }
}.

module SRr = {
 var rh : hash list
 var fr : from -> int
 proc init (r:int) = {
  rh <$ dlist dhash r;
  fr <$ difun r;
  QRO.fh <- fun x => nth witness rh (fr x);
 }

 quantum proc h {x:from} = {
  return (fr x, QRO.fh x);
 }
}.

module SROr : SRr_i = {
 proc init (r:int) = {
  SR.fr <$ difun r;
  QRO.init();
 }

 quantum proc h {x:from} = {
  return (SR.fr x, QRO.fh x);
 }
}.

axiom advantage_r q r (A<:AdvSRr{-SR, -QRO}):
 0 < r =>
 | Pr[IND_SRr(SROr,A).main(r) : res] - Pr[IND_SRr(SRr,A).main(r) : res] | <= 54 * q^3 / r.
```

5.6 Semi-Constant Distributions

Our formalization of semi-constant distributions follows the same style as that adopted in the previous sections.

Recall one can think of a semi-constant distribution as assigning a fixed value to the outputs of a subset X of hash inputs. All other input values are assigned outputs sampled from dhash. Subset X is implicitly sampled by using a Bernoulli distribution with bias λ to decide whether each input value is in X.

Our formalization is a strengthening of [55, Corollary 4.8], which we prove in Appendix B. In this stronger variant, the indistinguishability bound holds wrt to adversaries that may create a distinguishing advantage by causing a *classical bad* event based on the hidden set X. This bad event is defined by allowing the attacker to output (x, l), where x is a hash input and l contains up to k additional hash inputs.

This variant is tailored to our proof of the Full Domain Hash construction described in Section 5.6. In particular, the bad event corresponds to the case where x is an element of X and all the other values are not. We note that in the proof of security of FDH from semi-constant distributions given in [54], the same technique is used, but the proof is not modularized in a game-based assumption. The proof we give in Appendix B modularizes this argument into something that can be used more generally in a black-box way.

```
op dbfun lam = dfun (fun _ => Biased.dbiased lam).

op good(X : from -> bool, x : from, l : from list) = X x /\ X \cap l = {} /\ size l <= k.

module SCD (A:AdvSCD) = {
 proc F0(lam:real) = {
  X <$ dfun (dbiased lam);
  QRO.init();
  y <$ dhash;
  (c,x,l) <- A(QRO).main();
  return (c /\ good X x l);
 }

 proc F1(lam:real) = {
  X <$ dbfun lam;
  QRO.init();
  y <$ dhash;
  QRO.hf <- fun m => if X m then y else QRO.h m;
  (c,x,l) <- A(QRO).main();
  return (c /\ good X x l);
 }
}.

axiom advantage λ (A<:AdvSCD{-SCD,-QRO}) : 0 <= λ <= 1 =>
 | Pr[SCD(A).F0(λ) : res] - Pr[SCD(A).F1(λ) : res] | <= (2q+k+1)/6 λ^2.
```

6 CASE STUDIES

In this section, we briefly describe all case studies we have mechanized in EasyPQC.

6.1 PRF-based MACs

In cryptography, a message authentication code (MAC) is a short piece of information used to ensure that the message comes from the claimed sender and has not been changed. A MAC system consists of two algorithms: a (possibly) randomized MAC signing algorithm $\mathsf{sign}(k, m)$ and a verification algorithm $\mathsf{verify}(k, m, t)$. Here k denotes the secret key, m denotes a message and t is the MAC tag for message m. Classically, a MAC system is secure if no adversary can forge a valid tag of an unqueried message,when the adversary is given oracle access to the signing algorithm.

In the quantum setting, the adversary is allowed to maintain its own quantum state and issue quantum queries to the signing oracle. Let $\sum_{m,x,y} \psi_{m,x,y} |m, x, y\rangle$ be the adversary's state prior to

issuing a signing query. The signing oracle transforms this state as

$$\sum_{m,x,y} \psi_{m,x,y}|m,x,y\rangle \rightarrow \sum_{m,x,y} \psi_{m,x,y}|m,x \oplus \mathsf{sign}(k,m),y\rangle$$

The plus-one (PO) security of MAC against quantum adversaries is defined in [17] as: For any quantum adversary, we say the MAC system is PO-secure, if the adversary cannot output $(q+1)$ distinct and valid pairs of message and tag after issuing q quantum queries to the signing oracle.

The construction. The PO-secure MAC construction [17] we mechanized in EasyPQC is based on quantum secure pseudorandom function (PRF) [53], which is a PRF that remains indistinguishable from a random function even when the adversary can issue quantum queries to the PRF. The tag of message m is the output of PRF for input m, i.e. $\mathsf{sign}(k,m) = \mathsf{PRF}(k,m)$. The security proof can be formalized as follows: In the only game-hop, the PRF is replaced by a truly random function of the same domain and range. Because of the quantum security of the PRF, this game-hop is indistinguishable to the quantum adversary. After this game-hop, the adversary needs to produce $(q+1)$ input/output pairs after q quantum queries to a random oracle. This success probability is bounded by $(q+1)/N$, where N is the size of the range of the random oracle. This bound is shown in [17] and we encode it as an axiom in EasyPQC. The final statement of the theorem is as follows:

```
lemma conclusion :
  let ϵ =
    |Pr[IND_PRF(PRF, B(A)).main : res] - Pr[IND_PRF(RO, B(A)).main : res]| in
  Pr[EUF_qMAC(A,QMAC).main : res] ≤ ϵ + (q+1)/N.
```

6.2 Identity-Based Encryption

An identity-based encryption (IBE) [16, 41] is a generalization of public key encryption. In IBE, the encryption of message m and the generation of secret key are associated with an identity string id (or id'). The secret key can be used to recover m from the ciphertext if their associated identities are equal. We say the IBE scheme is secure, if the semantic security for the challenge identity (of adversary's choice) holds even if the adversary can obtain secret keys for any other identity. In [54], Zhandry showed that the GPV-IBE scheme [26] from lattices can be proven secure in the QROM. We have formalized this result in EasyPQC as follows.

The construction. There are three components in the GPV-IBE: a hash function H, a pre-image samplable function (PSF) and an encryption scheme $\mathsf{E} = (\mathsf{E.Enc}, \mathsf{E.Dec})$. A PSF is a quadruple of algorithms $\mathsf{F} = (\mathsf{Gen}, \mathsf{Sample}, \mathsf{f}, \mathsf{f}^{-1})$, where Gen generates private/public key pairs $(\mathsf{msk}, \mathsf{mpk})$ and f_{mpk} is a function. Sample samples x from a distribution D such that $f_{\mathsf{mpk}}(x)$ is uniform, and $f^{-1}_{\mathsf{msk}}(y)$ samples from D conditioned on $f_{\mathsf{mpk}}(x) = y$. These two ways of generating distribution $\{(x,y)\}$ are indistinguishable based on lattices.

The construction takes H to map identities to the co-domain of the PSF, which in turn matches the public key space for the encryption scheme. The preimage sampling algorithm allows obtaining a good secret key for the PKE conditioned on a given public key. The GPV-IBE scheme $\mathsf{IBE} = (\mathsf{keygen}, \mathsf{Extract}^H, \mathsf{Enc}^H, \mathsf{Dec})$ can then be briefly sketched as follows: 1) master key generation is simply the key generation for the PSS; 2) extraction is defined as $\mathsf{Extract}^H(\mathsf{msk}, \mathsf{id}) = \mathsf{f}^{-1}_{\mathsf{msk}}(\mathsf{H}(\mathsf{id}))$, i.e., a PKE secret key sampled conditionally on a of a PKE public key; 3) encryption and decryption just use the public-key encryption scheme as expected.

The result. An IBE scheme is said to be post-quantum secure in the QROM if a QPT adversary cannot break the semantic security of a challenge identity, even with access to an oracle that reveals secret keys for other identities of choice. Here, access to the oracles in the game is classical, but access to the QROM is a quantum interaction. The security result proved in EasyPQC states that the advantage of any quantum adversary in this game (IDCPA_QROM) is bounded by the post-quantum security of the PKE scheme.

```
lemma conclusion :
  let ϵ = |Pr[IDCPA_QROM(A,GPV(E)) : res] - 1/2| in
  let λ = 2ϵ^2/(2q_h+7q_e+3) in
  ϵ^2/(2q_s+7q_e+3) ≤ |Pr[CPA(B(A), ES(E)).main(λ) : res]- 1/2|.
```

Here q_h and q_e are the number of queries placed by the quantum attacker to the QROM and extraction oracle, respectively, and E is any public-key encryption scheme that can be transformed into an IBE using the GPV functor. (The ES functor modifies the PKE syntactically to rely on the PSF for key generation).

The proof. The EasyCrypt formalization of the proof follows closely the proof in [54] and uses the library for semi-constant distributions introduced in Section 5.2.

Intuitively, the proof proceeds as a sequence of games that first modifies the IBE security game to sample a subset X of the identity space, where one guesses that the challenge identity will reside, but none of the other extrated identities will. This multiplicative hop introduces a loss in advantage which bound by the probability of a successful guess which can be proved exactly the same techniques as for any classical proof: note that the guess is outside of the adversary's view, and that extraction queries are classical.

The implementation of the random oracle is then modified using the semi-constant distribution theory introduced in Section 5.2: for all identities in X, the output of the hash function is fixed to the same randomly sampled public-key. We construct a quantum attacker that breaks this assumption if the IBE attacker can detect the modification in the game. Here we rely crucially on the stronger variant of the semi-constant distribution result that allows the adversary to embed a bad event in the distinguishing probability (this matches the correct guess event previously introduced in the game). The extended logics for dealing with abstract quantum adversaries is important in this step, as well as in the reduction that follows.

Indeed, the proof then concludes in the expected way: use the properties of the PFS to construct a reduction to the security of the public-key encryption scheme. As a final comment, note that the result given in the bound fixes a concrete value for λ, which minimizes a general bound obtained for any λ in the range $(0,1)$. We do not prove that this value gives the minimum.

6.3 Full Domain Hash Signatures

We have introduced Full-Domain-Hash signatures in Section 3.1. There we described a proof that relies on the claw-free property of the trapdoor permutation. Here we describe how we proved two

additional results wrt the security of this construction. We give a brief description of the first result, since it follows very closely from a combination of the proof in Section 3.1 and the proof of the IBE construction we gave in the previous section. We then give a more detailed description of the proof based on small-range distributions, in which we tried to transpose the proof given in [55] but failed. It turned out that the proof was not correct and needed fixing.

A proof using semi-constant distributions. The proof of security given in Section 3.1 can be modified to obtain a reduction to the one-wayness property of the trapdoor permutation as described in [54]. We formalized this alternative proof as follows. We reuse the initial game-hop where the challenger guesses the set X in which the message chosen for the forgery will reside, but all the messages queried to the signature oracle do not. Note that this hop conceptually is very similar to the one used in the previous subsection for the IBE scheme, where the target identity corresponds to the message used in the forgery, and signature queries correspond to extraction queries. The next hop is therefore identical to the semi-constant distribution reduction given above: we modify the QROM using the library described in Section 5.2. At this point the output of the QROM for all values in X can be set to a given random value y, which allows a reduction to the one-wayness property of the trapdoor-permutation: the forged signature will be a preimage of y. We conclude the proof by writing this reduction explicitly. The final statement of the theorem is as follows.

lemma conclusion :
let $\lambda = \frac{1}{qs+1}$ **in**
Pr[EUF_QROM(A,FDH) : **res**] $\leq \frac{\Pr[\mathrm{OW(B(A))} : \mathrm{res}]}{\lambda(1-\lambda)qs} + \frac{(2q_h+3qs+3)\lambda}{6(1-\lambda)qs}$.

A new proof using small-range distributions. Zhandry [55] gives a proof of the security of the Full Domain Hash signature scheme with an improved bound using the small-range distribution technique. The proof goes as follows:

(1) Make a syntactic change that modifies the random oracle to first use a different random function O to compute a preimage to the trapdoor permutation (rather than an image), and then compute the trapdoor permutation.
(2) Use the small-range distribution assumption to change O so that it maps to a small set of r values, each of them sampled uniformly at random. Here r is chosen so that a successful attacker, say with advantage ϵ, retains at least $\epsilon/2$ advantage (r is fixed as a function of ϵ).
(3) Modify the game once more to sample a value i^* in the range $[0, r)$ and define a bad event if the following conditions do not hold: i. $O(m^*) = i^*$, where m^* is the message used in the forgery; and ii. forall m queried to the signing oracle, $O(m) \neq i^*$.
(4) Bound the bad event using a purely probabilistic argument.
(5) Show that any advantage retained by the adversary in this final game can be reduced to the one-wayness of the trapdoor permutation by fixing the OW challenge y to be the value of the hash function on m^*.

We failed when formalizing step 4 in the proof above: we could not bound the probability of bad occurring using only a probabilistic argument. The problem is that, although the choice of i^* is information-theoretically hidden from the adversary, the output of the O oracle is not. In particular, an adversary could choose an m^* that causes bad to occur because $H(m^*) = H(m)$ for some m queried to the signature oracle.

We conclude the proof by introducing an additional game-hop at step (4) where we exclude the possibility of collisions using the collision finding assumption we described in Section 5.2. We then are able to conclude the proof as before. Our final statement of this proof is as follows.

lemma conclusion :
let $k = 54(q+2)^3$ **in**
let ϵ = **Pr**[EUF_QROM(A,FDH) : **res**] **in**
let r = $3\lfloor \frac{k}{\epsilon} \rfloor$ **in**
$\frac{\epsilon^2}{6k} \leq$ **Pr**[OW(B(A)).main(r) : **res**].

As a follow up to the previous proof, and building on the insights we obtained, we formalized a simpler version, which is more natural and follows the general sequence of games used for semi-constant distributions; it goes as follows. There is an initial guessing step, which is performed independently of the view of the adversary. This allows us to bound the probability of the bad event identified in step for (4) above. We then apply the stronger variant of the small-range distribution transformation described in Section 5.2. The reduction to the one-wayness property then works as before.

7 RELATED WORK

Program logics for quantum programs. Quantum Hoare logic has become an active field in recent years, aiming to verify genuinely quantum properties. [40] gave the first rigorous semantics of a quantum language QPL. Later on, [19] established an Ensemble Exogenous Quantum Propositional Logic (EEQPL) and [31] proposed a quantitative quantum Hoare logic; both are designed for classical-quantum programming languages, but difficult to use in practice [37]. [23] suggested using Hermitian operators as quantum assertions and opened up the way of expectations-based verification. Based on this notion, [50] established the first sound and relatively complete quantum Hoare logic but only considered quantum variables. This work has been promoted [25] and many variants have been proposed [30, 52, 57, 58]. Besides, a quantum Hoare logic with ghost variables is introduced in [45] which is useful for dealing with mixed states/distributions. Practical tools have also been developed in recent years, including [18, 35] in Isabelle/HOL, [27, 28, 39] in Coq; all of them are based on matrix representation and calculations. They show that deriving and verifying general quantum properties are relatively difficult in implementation (with unexpected times and length of codes), for example, verifying Grover's search algorithm needs around 770 lines of code and one-person week in [27] and over 3000 lines in [35].

There are closely related works that established quantum relational Hoare logic for reasoning about the relational properties of quantum programs [33, 46, 47] and [10]. We point out that our core logic is not a fragment of them; there is an essential difference in design decisions. All [10, 33, 46, 47] focus on deriving general quantum properties and thus face the common difficulty in use; besides, they interpret predicates as physical observables/projections(subspaces) which is not compatible with first-order logic, e.g., the implication has no counterpart in quantum logic [22], which prevents generalizing current tools for classical verification such as

weakest precondition calculus to the quantum setting. In contrast, by solving the technical problem of CM (closed-under-mixture) condition, we use first-order predicate logic as assertion logic which enables the reuse of tools in EasyCrypt.

Computer-aided cryptography. Computer-aided cryptography [3] is an active area of research. However, (post-)quantum cryptography has received limited attention in this context. One notable tool is AutoLWE [7], an automated for proving security of post-quantum constructions based on Learning With Errors. However, the underlying theory is restricted to classical adversaries, and to the standard model. Hirschi [29] develop symbolic techniques for quantum protocols. However, his approach is more focused on attack detection. Given the absence of tools for post-quantum proofs, a natural question is whether existing tools are sound w.r.t. quantum adversaries. While the question is out of scope of the paper, we note that some computational tools such as CryptoVerif [14] or Squirrel [2] do not represent adversarial computations explicitly, and thus other techniques will be required to prove their soundness against quantum adversaries.

8 OTHER PROOF TECHNIQUES FOR QROM

In addition to the small-range and semi-constant techniques that are supported in EasyPQC, we discuss additional QROM techniques, such as compressed oracle [56] and one-way to hiding lemma [44] that can potentially be supported by extending the EasyPQC system.

The one-way to hiding (O2H) lemma, introduced by Unruh [1, 44], is another useful tool for proving security in the QROM. In a nutshell, the O2H lemma and its variants state that a random oracle is indistinguishable from a similar function, which is evaluated to different outputs only on a small random subset of inputs. The analysis of the performance of any adversary given access to the similar function leads to a bound in the real game where a random oracle is queried. We can axiomatize O2H results in a similar way to what was done for semi-constant and small-range distributions, as long as these results are used to hop between games whose code does not perform measurements. Indeed, we could formalize variant (1) of the O2H theorem in [1] even though it relies on a semi-classical oracle, by using that form stated in [13]: since the semi-classical oracle is only used to define an event that is used to bound the distance between the two games, we can state the axiom by moving the measurement step inside the adversary and redefining that event over the adversary's output. However, formalizing proofs that explicitly manipulate semi-classical oracles would require extending the tool to allow capturing the measurement operation.

The compressed oracle technique, introduced by Zhandry [56], has shown its powerfulness to prove quantum security in the QROM and quantum query complexity lower bounds. The technique resembles the idea of lazy sampling for classical random oracles [20]. The initial superposition of random functions, before any query is made, is a tensor product of zero states in the Fourier basis, and can be viewed as a quantum analogue of an empty database. After a query is made, one random entry gets "recorded" into the "database" in superposition. The performance of a quantum algorithm can be analyzed by keep tracking of the recorded entries.

The description of the compressed oracle technique is currently unsupported in EasyPQC, as applying them requires matrix representation and calculations in a low level. It is relatively straightforward to extend our core logic by adding projections (subspaces) as quantum atomic formulas and establish rules for other quantum constructs (initialization, general unitary transformation and measurement). On the other hand, while providing such an extension will make EasyCrypt more expressive, the implementation will incur a substantial increase of complexity. Thus it remains an interesting future line of research to explore the possibility of supporting proofs using the techniques without in-depth matrix calculations.

9 CONCLUSION

EasyPQC is a variant of EasyCrypt that supports reasoning about security of post-quantum constructions in the QROM. Currently, EasyPQC only supports the semi-constant and small-range techniques, but we plan to provide support for more advanced QROM techniques. In parallel, we intend to use EasyPQC to formalize security proofs for NIST Post-Quantum Competitions finalists. A longer-term goal is to give a mechanized security proofs for post-quantum variants of the TLS protocol.

ACKNOWLEDGEMENT

The work of Xiong Fan and Shih-Han Hung were partially performed while the authors were at University of Maryland. Xiaodi Wu is supported by AFOSR Young Investigator Program (YIP) Award (FA95502110094) and NSF CAREER Award (NSF-CCF-1942837). We thank Chris Hawblitzel and the anonymous reviewers for improving our work.

REFERENCES

[1] Andris Ambainis, Mike Hamburg, and Dominique Unruh. 2019. Quantum Security Proofs Using Semi-classical Oracles. In *CRYPTO 2019, Part II (LNCS)*, Alexandra Boldyreva and Daniele Micciancio (Eds.), Vol. 11693. Springer, Heidelberg, 269–295. https://doi.org/10.1007/978-3-030-26951-7_10

[2] David Baelde, Stéphanie Delaune, Charlie Jacomme, Adrien Koutsos, and Solène Moreau. 2021. An Interactive Prover for Protocol Verification in the Computational Model. In *IEEE S&P 2021 - 42nd IEEE Symposium on Security and Privacy*. San Fransisco / Virtual, United States, t.b.d.

[3] Manuel Barbosa, Gilles Barthe, Karthik Bhargavan, Bruno Blanchet, Cas Cremers, Kevin Liao, and Bryan Parno. 2019. SoK: Computer-Aided Cryptography. Cryptology ePrint Archive, Report 2019/1393. (2019). https://eprint.iacr.org/2019/1393.

[4] Manuel Barbosa, Gilles Barthe, Benjamin Grégoire, Adrien Koutsos, and Pierre-Yves Strub. 2021. Mechanized Proofs of Adversarial Complexity and Application to Universal Composability. *IACR Cryptol. ePrint Arch.* 2021 (2021), 156. https://eprint.iacr.org/2021/156

[5] Gilles Barthe, François Dupressoir, Benjamin Grégoire, César Kunz, Benedikt Schmidt, and Pierre-Yves Strub. 2013. EasyCrypt: A Tutorial. In *Foundations of Security Analysis and Design VII - FOSAD 2012/2013 Tutorial Lectures (Lecture Notes in Computer Science)*, Alessandro Aldini, Javier López, and Fabio Martinelli (Eds.), Vol. 8604. Springer, 146–166. https://doi.org/10.1007/978-3-319-10082-1_6

[6] Gilles Barthe, Thomas Espitau, Benjamin Grégoire, Justin Hsu, Léo Stefanesco, and Pierre-Yves Strub. 2015. Relational Reasoning via Probabilistic Coupling. In *Logic for Programming, Artificial Intelligence, and Reasoning - 20th International Conference, LPAR-20 2015, Suva, Fiji, November 24-28, 2015, Proceedings (Lecture Notes in Computer Science)*, Martin Davis, Ansgar Fehnker, Annabelle McIver, and Andrei Voronkov (Eds.), Vol. 9450. Springer, 387–401. https://doi.org/10.1007/978-3-662-48899-7_27

[7] Gilles Barthe, Xiong Fan, Joshua Gancher, Benjamin Grégoire, Charlie Jacomme, and Elaine Shi. 2018. Symbolic Proofs for Lattice-Based Cryptography. In *ACM CCS 2018*, David Lie, Mohammad Mannan, Michael Backes, and XiaoFeng Wang (Eds.). ACM Press, 538–555. https://doi.org/10.1145/3243734.3243825

[8] Gilles Barthe, Benjamin Grégoire, and Santiago Zanella Béguelin. 2009. Formal certification of code-based cryptographic proofs. In *Proceedings of the 36th ACM*

SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL 2009, Savannah, GA, USA, January 21-23, 2009, Zhong Shao and Benjamin C. Pierce (Eds.). ACM, 90–101. https://doi.org/10.1145/1480881.1480894

[9] Gilles Barthe, Benjamin Grégoire, Sylvain Heraud, and Santiago Zanella Béguelin. 2011. Computer-Aided Security Proofs for the Working Cryptographer. In *CRYPTO 2011 (LNCS)*, Phillip Rogaway (Ed.), Vol. 6841. Springer, Heidelberg, 71–90. https://doi.org/10.1007/978-3-642-22792-9_5

[10] Gilles Barthe, Justin Hsu, Mingsheng Ying, Nengkun Yu, and Li Zhou. 2019. Relational Proofs for Quantum Programs. *Proc. ACM Program. Lang.* 4, POPL, Article 21 (Dec. 2019), 29 pages. https://doi.org/10.1145/3371089

[11] Mihir Bellare and Phillip Rogaway. 1993. Random Oracles are Practical: A Paradigm for Designing Efficient Protocols. In *ACM CCS 93*, Dorothy E. Denning, Raymond Pyle, Ravi Ganesan, Ravi S. Sandhu, and Victoria Ashby (Eds.). ACM Press, 62–73. https://doi.org/10.1145/168588.168596

[12] Mihir Bellare and Phillip Rogaway. 1996. The Exact Security of Digital Signatures: How to Sign with RSA and Rabin. In *EUROCRYPT'96 (LNCS)*, Ueli M. Maurer (Ed.), Vol. 1070. Springer, Heidelberg, 399–416. https://doi.org/10.1007/3-540-68339-9_34

[13] Nina Bindel, Mike Hamburg, Kathrin Hövelmanns, Andreas Hülsing, and Edoardo Persichetti. 2019. Tighter Proofs of CCA Security in the Quantum Random Oracle Model. In *TCC 2019, Part II (LNCS)*, Dennis Hofheinz and Alon Rosen (Eds.), Vol. 11892. Springer, Heidelberg, 61–90. https://doi.org/10.1007/978-3-030-36033-7_3

[14] Bruno Blanchet. 2006. A Computationally Sound Mechanized Prover for Security Protocols. In *2006 IEEE Symposium on Security and Privacy*. IEEE Computer Society Press, 140–154. https://doi.org/10.1109/SP.2006.1

[15] Dan Boneh, Özgür Dagdelen, Marc Fischlin, Anja Lehmann, Christian Schaffner, and Mark Zhandry. 2011. Random Oracles in a Quantum World. In *ASIACRYPT 2011 (LNCS)*, Dong Hoon Lee and Xiaoyun Wang (Eds.), Vol. 7073. Springer, Heidelberg, 41–69. https://doi.org/10.1007/978-3-642-25385-0_3

[16] Dan Boneh and Matthew K. Franklin. 2001. Identity-Based Encryption from the Weil Pairing. In *CRYPTO 2001 (LNCS)*, Joe Kilian (Ed.), Vol. 2139. Springer, Heidelberg, 213–229. https://doi.org/10.1007/3-540-44647-8_13

[17] Dan Boneh and Mark Zhandry. 2013. Quantum-Secure Message Authentication Codes. In *EUROCRYPT 2013 (LNCS)*, Thomas Johansson and Phong Q. Nguyen (Eds.), Vol. 7881. Springer, Heidelberg, 592–608. https://doi.org/10.1007/978-3-642-38348-9_35

[18] Anthony Bordg, Hanna Lachnitt, and Yijun He. 2020. Certified Quantum Computation in Isabelle/HOL. *Journal of Automated Reasoning* (24 Dec 2020). https://doi.org/10.1007/s10817-020-09584-7

[19] Rohit Chadha, Paulo Mateus, and Amílcar Sernadas. 2006. Reasoning about imperative quantum programs. *Electronic Notes in Theoretical Computer Science* 158 (2006), 19–39. https://doi.org/10.1016/j.entcs.2006.04.003

[20] Kai-Min Chung, Serge Fehr, Yu-Hsuan Huang, and Tai-Ning Liao. 2021. On the Compressed-Oracle Technique, and Post-Quantum Security of Proofs of Sequential Work. (2021). arXiv:quant-ph/2010.11658

[21] Jean-Sébastien Coron. 2000. On the Exact Security of Full Domain Hash. In *CRYPTO 2000 (LNCS)*, Mihir Bellare (Ed.), Vol. 1880. Springer, Heidelberg, 229–235. https://doi.org/10.1007/3-540-44598-6_14

[22] Maria Luisa Dalla Chiara. 1986. Quantum logic. In *Handbook of philosophical logic*. Springer, 427–469.

[23] ELLIE D'HONDT and PRAKASH PANANGADEN. 2006. Quantum weakest preconditions. *Mathematical Structures in Computer Science* 16, 3 (2006), 429–451. https://doi.org/10.1017/S0960129506005251

[24] Jelle Don, Serge Fehr, Christian Majenz, and Christian Schaffner. 2019. Security of the Fiat-Shamir Transformation in the Quantum Random-Oracle Model. In *CRYPTO 2019, Part II (LNCS)*, Alexandra Boldyreva and Daniele Micciancio (Eds.), Vol. 11693. Springer, Heidelberg, 356–383. https://doi.org/10.1007/978-3-030-26951-7_13

[25] Yuan Feng and Mingsheng Ying. 2020. Quantum Hoare logic with classical variables. (2020). arXiv:cs.LO/2008.06812

[26] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. 2008. Trapdoors for hard lattices and new cryptographic constructions. In *40th ACM STOC*, Richard E. Ladner and Cynthia Dwork (Eds.). ACM Press, 197–206. https://doi.org/10.1145/1374376.1374407

[27] Kesha Hietala, Robert Rand, Shih-Han Hung, Liyi Li, and Michael Hicks. 2020. Proving Quantum Programs Correct. (2020). arXiv:cs.PL/2010.01240

[28] Kesha Hietala, Robert Rand, Shih-Han Hung, Xiaodi Wu, and Michael Hicks. 2021. A Verified Optimizer for Quantum Circuits. *Proc. ACM Program. Lang.* 5, POPL, Article 37 (Jan. 2021), 29 pages. https://doi.org/10.1145/3434318

[29] Lucca Hirschi. 2019. Symbolic Abstractions for Quantum Protocol Verification. *CoRR* abs/1904.04186 (2019). arXiv:1904.04186 http://arxiv.org/abs/1904.04186

[30] Shih-Han Hung, Kesha Hietala, Shaopeng Zhu, Mingsheng Ying, Michael Hicks, and Xiaodi Wu. 2019. Quantitative Robustness Analysis of Quantum Programs. *Proc. ACM Program. Lang.* 3, POPL, Article 31 (Jan. 2019), 29 pages. https://doi.org/10.1145/3290344

[31] Yoshihiko Kakutani. 2009. A logic for formal verification of quantum programs. In *Proceedings of the 13th Asian conference on Advances in Computer Science: information Security and Privacy (ASIAN 2009)*, Anupam Datta (Ed.). Springer, Springer Berlin Heidelberg, Berlin, Heidelberg, 79–93. https://doi.org/10.1007/978-3-642-10622-4_7

[32] Burkhard Kümmerer and Kay Schwieger. 2016. Diagonal couplings of quantum Markov chains. *Infinite Dimensional Analysis, Quantum Probability and Related Topics* 19, 2 (2016), 1650012.

[33] Yangjia Li and Dominique Unruh. 2019. Quantum Relational Hoare Logic with Expectations. (2019). arXiv:cs.LO/1903.08357

[34] Torgny Lindvall. 2002. *Lectures on the coupling method*. Courier Corporation.

[35] Junyi Liu, Bohua Zhan, Shuling Wang, Shenggang Ying, Tao Liu, Yangjia Li, Mingsheng Ying, and Naijun Zhan. 2019. Formal Verification of Quantum Algorithms Using Quantum Hoare Logic. In *Computer Aided Verification*, Isil Dillig and Serdar Tasiran (Eds.). Springer International Publishing, Cham, 187–207. https://doi.org/10.1007/978-3-030-25543-5_12

[36] Qipeng Liu and Mark Zhandry. 2019. Revisiting Post-quantum Fiat-Shamir. In *CRYPTO 2019, Part II (LNCS)*, Alexandra Boldyreva and Daniele Micciancio (Eds.), Vol. 11693. Springer, Heidelberg, 326–355. https://doi.org/10.1007/978-3-030-26951-7_12

[37] Robert Rand. 2019. Verification Logics for Quantum Programs. (2019). arXiv:cs.LO/1904.04304

[38] Oded Regev. 2005. On lattices, learning with errors, random linear codes, and cryptography. In *37th ACM STOC*, Harold N. Gabow and Ronald Fagin (Eds.). ACM Press, 84–93. https://doi.org/10.1145/1060590.1060603

[39] Steve Zdancewic Robert Rand, Jennifer Paykin. 2017. QWIRE Practice: Formal Verification of Quantum Circuits in Coq. In *14th International Conference on Quantum Physics and Logic 2017 (QPL '17)*. https://qpl.science.ru.nl/papers/QPL_2017_paper_45.pdf

[40] PETER SELINGER. 2004. Towards a quantum programming language. *Mathematical Structures in Computer Science* 14, 4 (2004), 527–586. https://doi.org/10.1017/S0960129504004256

[41] Adi Shamir. 1984. Identity-based cryptosystems and signature schemes. In *Workshop on the theory and application of cryptographic techniques*. Springer, 47–53.

[42] Fang Song. 2014. A Note on Quantum Security for Post-Quantum Cryptography. In *Post-Quantum Cryptography - 6th International Workshop, PQCrypto 2014, Waterloo, ON, Canada, October 1-3, 2014. Proceedings (Lecture Notes in Computer Science)*, Michele Mosca (Ed.), Vol. 8772. Springer, 246–265. https://doi.org/10.1007/978-3-319-11659-4_15

[43] Hermann Thorisson. 2000. *Coupling, Stationarity, and Regeneration*. springer.

[44] Dominique Unruh. 2015. Revocable quantum timed-release encryption. *Journal of the ACM (JACM)* 62, 6 (2015), 1–76.

[45] Dominique Unruh. 2019. Quantum Hoare Logic with Ghost Variables. In *2019 34th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*. 1–13. https://doi.org/10.1109/LICS.2019.8785779

[46] Dominique Unruh. 2019. Quantum Relational Hoare Logic. *Proc. ACM Program. Lang.* 3, POPL, Article 33 (Jan. 2019), 31 pages. https://doi.org/10.1145/3290346

[47] Dominique Unruh. 2020. Local Variables and Quantum Relational Hoare Logic. (2020). arXiv:cs.LO/2007.14155

[48] Cédric Villani. 2008. *Optimal transport: Old and new*. springer.

[49] Andreas Winter. 2016. Tight uniform continuity bounds for quantum entropies: conditional entropy, relative entropy distance and energy constraints. *Communications in Mathematical Physics* 347, 1 (2016), 291–313.

[50] Mingsheng Ying. 2011. Floyd–hoare logic for quantum programs. *ACM Transactions on Programming Languages and Systems (TOPLAS)* 33, 6, Article 19 (2011), 49 pages. https://doi.org/10.1145/2049706.2049708

[51] Mingsheng Ying. 2016. *Foundations of quantum programming*. Morgan Kaufmann.

[52] Mingsheng Ying, Li Zhou, and Yangjia Li. 2018. Reasoning about Parallel Quantum Programs. (2018). arXiv:cs.LO/1810.11334

[53] Mark Zhandry. 2012. How to Construct Quantum Random Functions. In *53rd FOCS*. IEEE Computer Society Press, 679–687. https://doi.org/10.1109/FOCS.2012.37

[54] Mark Zhandry. 2012. Secure Identity-Based Encryption in the Quantum Random Oracle Model. In *CRYPTO 2012 (LNCS)*, Reihaneh Safavi-Naini and Ran Canetti (Eds.), Vol. 7417. Springer, Heidelberg, 758–775. https://doi.org/10.1007/978-3-642-32009-5_44

[55] Mark Zhandry. 2015. *Cryptography in the Age of Quantum Computers*. Ph.D. Dissertation. Stanford University. https://www.cs.princeton.edu/~mzhandry

[56] Mark Zhandry. 2019. How to Record Quantum Queries, and Applications to Quantum Indifferentiability. In *CRYPTO 2019, Part II (LNCS)*, Alexandra Boldyreva and Daniele Micciancio (Eds.), Vol. 11693. Springer, Heidelberg, 239–268. https://doi.org/10.1007/978-3-030-26951-7_9

[57] Li Zhou, Gilles Barthe, Justin Hsu, Mingsheng Ying, and Nengkun Yu. 2021. A Quantum Interpretation of Bunched Logic for Quantum Separation Logic. (2021). arXiv:cs.LO/2102.00329

[58] Li Zhou, Shenggang Ying, Nengkun Yu, and Mingsheng Ying. 2020. Strassen's theorem for quantum couplings. *Theoretical Computer Science* 802 (2020), 67–76. https://doi.org/10.1016/j.tcs.2019.08.026

A POST-QUANTUM RELATIONAL HOARE LOGIC, EXTENDED VERSION

In this section, we give all integrated materials for construct our core logic pqRHL as well as the proof of the soundness.

A.1 Quantum State

The *state space* of a quantum system is a Hilbert space $\mathcal{H}$, which is essentially a vector space with inner product in the finite-dimensional case. A *pure state* of the system is a unit column vector $|\psi\rangle \in \mathcal{H}$, and we use $\langle\psi|$ to denote its dual, i.e., conjugate transpose of $|\psi\rangle$. For example, the state space of a quantum bit (aka qubit) is a two-dimensional Hilbert space with basis states $|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$ and $|1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$, and any pure state of a qubit can be described in the form $\alpha|0\rangle + \beta|1\rangle = \begin{bmatrix} \alpha \\ \beta \end{bmatrix}$ satisfying normalization condition $|\alpha|^2 + |\beta|^2 = 1$. When the state is not completely known but could be in one of some pure states $|\psi_i\rangle$ with respective probabilities p_i, we call $\{(p_i, |\psi_i\rangle)\}$ an *ensemble* of pure states or a *mixed state*, and the system is fully described by the *density operator* $\rho = \sum_i p_i|\psi_i\rangle\langle\psi_i|$. For example, the completely mixed state of a qubit can be seen as ensemble $\{(0.5, |0\rangle), (0.5, |+\rangle)\}$ (i.e. the state is either $|0\rangle$ or $|+\rangle \triangleq \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ with the same probability 0.5) or density matrix

$$\frac{1}{2}|0\rangle\langle 0| + \frac{1}{2}|+\rangle\langle +| = \frac{1}{2}\begin{bmatrix} 1 \\ 0 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 \end{bmatrix} + \frac{1}{2}\begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix} \cdot \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} = \frac{1}{2}\begin{bmatrix} 1 & \\ & 0 \end{bmatrix} + \frac{1}{2}\begin{bmatrix} 1/2 & 1/2 \\ 1/2 & 1/2 \end{bmatrix} = \begin{bmatrix} 0.75 & 0.25 \\ 0.25 & 0.25 \end{bmatrix}$$

To indicate which system a state describes or an operation acts on, we use subscripts; for example, $\mathcal{H}_p$ is the state space of system p, $|0\rangle_p$ is the pure state $|0\rangle$ of the system p and $|1\rangle_q\langle 1|$ is the density matrix of the system q. The composite system is described by the tensor product of its subsystems; for example, a composite system pq with p, q being single qubit systems has the state space $\mathcal{H}_p \otimes \mathcal{H}_q$, and

$$|0\rangle_p \otimes |1\rangle_q = \begin{bmatrix} 1 \\ 0 \end{bmatrix}_p \otimes \begin{bmatrix} 0 \\ 1 \end{bmatrix}_q = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}_{pq}$$

(or, $|0\rangle_p|1\rangle_q$ for short) is a pure state in which subsystem p is in state $|0\rangle$ and subsystem q is in state $|1\rangle$. Due to the superposition principle, there exist states like

$$|\Phi\rangle_{pq} = \frac{1}{\sqrt{2}}(|0\rangle_p|0\rangle_q + |1\rangle_p|1\rangle_q) = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix}_{pq} \tag{1}$$

that cannot be written in the simple tensor form $|\phi\rangle_p|\psi\rangle_q$, which are called *entangled states*. These states play a crucial role in applications of quantum computation and quantum communication.

The state of a composite system fully determines the state of each subsystem. Formally, given composite system pq in state ρ, subsystem q is then in state $\mathrm{tr}_p(\rho)$, where the partial trace $\mathrm{tr}_p(\cdot)$ over p is a mapping from operators on $\mathcal{H}_p \otimes \mathcal{H}_q$ to operators on $\mathcal{H}_q$ defined by:

$$\mathrm{tr}_p(|\phi_p\rangle_p\langle\psi_p| \otimes |\phi_q\rangle_q\langle\psi_q|) = \langle\psi_p|\phi_p\rangle \cdots |\phi_q\rangle_q\langle\psi_q|$$

for all $|\phi_p\rangle, |\psi_p\rangle \in \mathcal{H}_p$ and $|\phi_q\rangle, |\psi_q\rangle \in \mathcal{H}_q$ together with linearity. The state $\mathrm{tr}_q(\rho)$ of subsystem q can be defined symmetrically. We often use the notations $\rho|_p \triangleq \mathrm{tr}_q(\rho)$ and $\rho|_q \triangleq \mathrm{tr}_p(\rho)$ in order to explicitly indicate that $\rho|_p$ and $\rho|_q$ are states of p, q, respectively. For example, if the composite system pq is in state $|\Phi\rangle_{pq}$ defined in Eqn. 1 or equivalently represented by density operator Φ_{pq}

$$\Phi_{pq} = |\Phi\rangle_{pq}\langle\Phi| = \frac{1}{2}(|0\rangle_p\langle 0| \otimes |0\rangle_q\langle 0| + |0\rangle_p\langle 1| \otimes |0\rangle_q\langle 1| + |1\rangle_p\langle 0| \otimes |1\rangle_q\langle 0| + |1\rangle_p\langle 1| \otimes |1\rangle_q\langle 1|) = \frac{1}{2}\begin{bmatrix} 1 & & & 1 \\ & 0 & & \\ & & 0 & \\ 1 & & & 1 \end{bmatrix}_{pq} \tag{2}$$

then the partial traces $\Phi_{pq}\big|_q = \mathrm{tr}_p(|\Phi\rangle_{pq}\langle\Phi|) = \frac{1}{2}(|0\rangle_q\langle 0| + |1\rangle_q\langle 1|)$ and $\Phi_{pq}\big|_p = \mathrm{tr}_q(|\Phi\rangle_{pq}\langle\Phi|) = \frac{1}{2}(|0\rangle_p\langle 0| + |1\rangle_p\langle 1|)$ describe states of q and p, respectively.

For a set of quantum variables $V \subseteq \mathrm{qVar}$, its Hilbert space is denoted by $\mathcal{H}(V)$, and we use $\mathcal{D}(V)$ to denote the set of partial density operators on $\mathcal{D}(V)$. Suppose $V' \subseteq V$ and $\rho \in \mathcal{D}(V)$, then the state of subsystem V' is $\rho|_{V'} \triangleq \mathrm{tr}_{V-V'}(\rho)$ as defined above.

A.2 Classical-Quantum (cq) State

Let Var be the set of program classical variables and qVar the set of program quantum variables. Let $\Sigma \triangleq \mathrm{Var} \to \mathcal{V}$ be the set of classical states, where $\mathcal{V}$ for the set of values, including integers, booleans, etc.

The memory of a hybrid classical-quantum system is described by its classical part $\sigma \in \Sigma$ and quantum part $\rho \in \mathcal{D}(\mathrm{qVar})$, and we denoted by $\langle\sigma, \rho\rangle$. Since the execution of programs might be probabilistic, we need consider the (sub)distribution over such states. However, using

Classical	Probabilistic	Quantum	Classical-quantum
state	probability (sub)distribution	(partial) density operator	cq-state
$\sigma \in \Sigma \triangleq \mathrm{Var} \to \mathcal{V}$	$\mu \in \Sigma \to [0,1]$	$\rho \in \mathcal{D}(\mathrm{qVar})$	$\Delta \in \Sigma \to \mathcal{D}(\mathrm{qVar})$
	countable support		countable support

Table 1: Comparison of the basic notions in different language paradigms.

(sub)distributions over $\Sigma \times \mathcal{D}(\mathrm{qVar})$ suffers indistinguishably of quantum ensembles: different (sub)distributions can refer to the same state. To avoid such technical difficulty, [25] gives the following definition:

Definition A.1 (cq-states [25]). Given $V \subseteq \mathrm{qVar}$, a classical-quantum state Δ over V is a function in $\Sigma \to \mathcal{D}(V)$ such that

(1) the support of Δ, denoted $\lceil \Delta \rceil$, is countable. That is, $\Delta(\sigma) \neq \mathbf{0}$ for at most countably infinite many $\sigma \in \Sigma$;
(2) $\mathrm{tr}(\Delta) \triangleq \sum_{\sigma \in \lceil \Delta \rceil} \mathrm{tr}[\Delta(\sigma)] \leq 1$.

We use $\Xi(V)$ for the set of cq-states over V, i.e., $\Xi(V) \triangleq \Sigma \to \mathcal{D}(V)$; and $\Xi \triangleq \bigcup_{V \subseteq \mathrm{qVar}} \Xi(V)$. Further, we denote $\langle \sigma, \rho \rangle$ for singleton cq-state (or cq-memory), i.e., $\langle \sigma, \rho \rangle(\sigma') := \mathbb{I}(\sigma, \sigma') \cdot \rho$, where $\mathbb{I}$ is the indicator function and $\mathbb{I}(\sigma, \sigma') = 1$ iff $\sigma = \sigma'$. Then, for any cq-state, we can always write it as the summation of singleton cq-states, i.e., $\Delta = \sum_\sigma \langle \sigma, \Delta(\sigma) \rangle$, since

$$\forall\, \sigma'.\ \Delta(\sigma') = \sum_\sigma \mathbb{I}(\sigma, \sigma') \Delta(\sigma) = \Big(\sum_\sigma \langle \sigma, \Delta(\sigma) \rangle \Big)(\sigma').$$

In particular, $\langle \sigma, \mathbf{0} \rangle$ is the identity of $+$, i.e., $\langle \sigma, \mathbf{0} \rangle + \Delta = \Delta$.

We can further define the partial order $\sqsubseteq$ between two cq-states $\Delta, \Delta' \in \Xi(V)$ as:

$$\Delta \sqsubseteq \Delta' \triangleq \forall \sigma,\ \Delta(\sigma) \sqsubseteq \Delta'(\sigma),$$

the $\sqsubseteq$ in the RHS is the Löwner order of matrices, i.e., $A \sqsubseteq B$ if $B - A$ is positive semi-definite. [25] shows that $\Xi(V)$ is an ω-complete partial order set (CPO) under $\sqsubseteq$.

Table 1 gives a comparison of states in different language paradigms. For classical state, it is a mapping from variables to values. In probabilistic setting, the state is characterized by a probabilistic (sub)distribution over classical states. In quantum setting, the state is defined by the quantum state over all quantum variables, i.e., the (partial) density operator over Hilbert space $\mathcal{H}(\mathrm{qVar})$. Note that quantum variables might be entangled and cannot be fully characterized by the states over each quantum variable, thus it takes different notion than classical states. A classical-quantum state Δ can be regarded as a combination of probabilistic and quantum states: for any classical state σ, $\Delta(\sigma)$ gives the corresponding quantum states; note that the probability is now encoded in the quantum states [40], i.e., if the probability of obtaining σ is p and the corresponding (partial) quantum state is ρ, then we define $\Delta(\sigma) = p \cdot \rho$. Furthermore, $\mathrm{tr}[\Delta(\sigma)]$ gives the probability that the classical states is σ; that is why we have the condition $\mathrm{tr}[\Delta] \leq 1$, i.e., it should be a (sub)distribution.

It is straightforward to see how a cq-state Δ reduces to probabilistic/quantum state if we discard quantum/classical state, respectively. If we discard the quantum variables, physically it means that we trace out the quantum states, thus for any classical state σ, we obtain the probability $\mathrm{tr}[\Delta(\sigma)]$, or more formally, $\forall\, \sigma,\ \mu(\sigma) = \mathrm{tr}[\Delta(\sigma)]$ where μ is just the probabilistic (sub)distribution of the classical system. On the other hand, if we discard the classical variables, then we should sum up the quantum states over all possible classical states, i.e., $\sum_\sigma \Delta(\sigma)$ which yields the quantum state over qVar.

For simplicity, we define two functions $\pi_c : \Xi \to (\Sigma \to [0,1])$ and $\pi_q : \Xi \to \mathcal{D}$ to obtain the probability (sub)distribution and quantum state from a cq-state Δ by: for all Δ,

$$\pi_c(\Delta)(\sigma) = \mathrm{tr}[\Delta(\sigma)], \quad \pi_q(\Delta) = \sum_{\sigma \in \lceil \Delta \rceil} \Delta(\sigma).$$

Furthermore, we define the restriction of $\Delta \in \Xi(V)$ on $V' \subseteq V$ as:

$$\Delta|_{V'} = \sum_\sigma \langle \sigma,\ \Delta(\sigma)|_{V'} \rangle = \sum_\sigma \langle \sigma, \mathrm{tr}_{V - V'}[\Delta(\sigma)] \rangle$$

A.3 Classical-Quantum Coupling

Coupling is an essential concept to define validity of relational judgments. Probabilistic coupling, a standard abstraction from probability theory [34, 43, 48], is employed to interpret judgments in probabilistic relational logic [6, 8]; similar concept of quantum coupling has also been proposed [32, 49, 58] and used in establishing quantum relational logic [10, 33, 46, 47].

For two programs c_1 and c_2, we use tags $\langle 1 \rangle$ and $\langle 2 \rangle$ to indicate the first and the second program, e.g., for the program variables of c_1, we renamed them by tagging with $\langle 1 \rangle$, for the program variables of c_2, we renamed them by tagging with $\langle 2 \rangle$. We also define:

$$\mathrm{Var}\langle 1 \rangle := \{x\langle 1 \rangle \mid x \in \mathrm{Var}\}, \quad \mathrm{qVar}\langle 1 \rangle := \{q\langle 1 \rangle \mid q \in \mathrm{qVar}\},$$

$$\langle \mathsf{skip}, \sigma, \rho\rangle \to \langle E, \sigma, \rho\rangle \qquad \langle x := e, \sigma, \rho\rangle \to \langle E, \sigma[[\![e]\!]_\sigma/x], \rho\rangle$$

$$\frac{v \in \lceil [\![d]\!]_\sigma \rceil}{\langle x \xleftarrow{\$} d, \sigma, \rho\rangle \to \langle E, \sigma[v/x], [\![d]\!]_\sigma(v)\cdot\rho\rangle} \qquad \langle q := e, \sigma, \rho\rangle \to \langle E, \sigma, \textstyle\sum_{i=0}^{dq-1} |[\![e]\!]_\sigma\rangle_q\langle i|\rho|i\rangle_q\langle [\![e]\!]_\sigma|\rangle$$

$$\langle \overline{q} \mathrel{*}= U, \sigma, \rho\rangle \to \langle E, \sigma, ([\![U]\!]_\sigma)_{\overline{q}}\rho([\![U^\dagger]\!]_\sigma)_{\overline{q}}\rangle \qquad \frac{M = \{M_i\}_{i\in\mathcal{J}}}{\langle x := \mathsf{meas}\, M[\overline{q}], \sigma, \rho\rangle \to \langle E, \sigma[i/x], ([\![M_i]\!]_\sigma)_{\overline{q}}\rho([\![M_i^\dagger]\!]_\sigma)_{\overline{q}}\rangle}$$

$$\frac{\langle c_1, \sigma, \rho\rangle \to \langle c_1', \sigma', \rho'\rangle}{\langle c_1; c_2, \sigma, \rho\rangle \to \langle c_1'; c_2, \sigma', \rho'\rangle} \text{ where } E; c_2 \equiv c_2$$

$$\frac{\sigma \models b}{\langle \mathsf{if}\ e\ \mathsf{then}\ c\ \mathsf{else}\ c', \sigma, \rho\rangle \to \langle c, \sigma, \rho\rangle} \qquad \frac{\sigma \models \neg b}{\langle \mathsf{if}\ e\ \mathsf{then}\ c\ \mathsf{else}\ c', \sigma, \rho\rangle \to \langle c', \sigma, \rho\rangle}$$

$$\frac{\sigma \models \neg b}{\langle \mathsf{while}\ e\ \mathsf{do}\ c, \sigma, \rho\rangle \to \langle E, \sigma, \rho\rangle} \qquad \frac{\sigma \models b}{\langle \mathsf{while}\ e\ \mathsf{do}\ c, \sigma, \rho\rangle \to \langle c; \mathsf{while}\ e\ \mathsf{do}\ c, \sigma, \rho\rangle}$$

Table 2: Operational semantics for cq-statements. For distribution d, $\lceil d \rceil$ is support of d, i.e., $\lceil d \rceil := \{v : d(v) > 0\}$. E denotes the termination, i.e., the empty program.

and similar for $\langle 2\rangle$. The tagged classical states are $\Sigma\langle 1\rangle : \mathrm{Var}\langle 1\rangle \to \mathcal{V}$ and $\Sigma\langle 2\rangle : \mathrm{Var}\langle 2\rangle \to \mathcal{V}$, the product classical states

$$\Sigma_\times : \mathrm{Var}\langle 1\rangle \uplus \mathrm{Var}\langle 2\rangle \to \mathcal{V}.$$

For any $\sigma_\times \in \Sigma_\times$, it is convenient to write it as a pair $\sigma_\times = (\sigma_1; \sigma_2)$. Further, we define the product cq-state:

$$\Delta_\times : \Sigma_\times \to \mathcal{D}(\mathrm{qVar}\langle 1\rangle \cup \mathrm{qVar}\langle 2\rangle).$$

Now we are ready to define cq-coupling, which is a combination of probabilistic and quantum couplings, in the sense that probabilistic (resp. quantum) couplings are degenerate cases if we discard quantum (resp. classical) system.

Definition A.2 (cq-coupling). For any product cq-state $\Delta_\times$, its marginal function π_1 and π_2 are defined as follows:

$$\forall\, \sigma_1,\ \pi_1(\Delta_\times)(\sigma_1) \triangleq \sum_{\sigma_2} \mathrm{tr}_{\mathrm{qVar}\langle 2\rangle}[\Delta_\times(\sigma_1; \sigma_2)]$$

$$\forall\, \sigma_2,\ \pi_2(\Delta_\times)(\sigma_2) \triangleq \sum_{\sigma_1} \mathrm{tr}_{\mathrm{qVar}\langle 1\rangle}[\Delta_\times(\sigma_1; \sigma_2)]$$

We say $\Delta_\times$ is a cq-coupling of (Δ_1, Δ_2) if its marginals are Δ_1 and Δ_2, i.e.,

$$\pi_1(\Delta_\times) = \Delta_1, \quad \pi_2(\Delta_\times) = \Delta_2.$$

Example A.3 (identity coupling). Suppose $\Delta(x = i) = \frac{1}{2}|i\rangle\langle i|$ for $i = 0, 1$, then product cq-state

$$\Delta_\times(x\langle 1\rangle = i, x\langle 2\rangle = i) = \frac{1}{2}|i\rangle\langle i| \otimes |i\rangle\langle i|$$

is a cq-coupling of (Δ, Δ).

If we apply π_c function to discard quantum states, it is straightforward to see that if $\Delta_\times$ is a cq-coupling of (Δ_1, Δ_2), then $\pi_c(\Delta_\times)$ is a probabilistic coupling of $(\pi_c(\Delta_1), \pi_c(\Delta_2))$. Similar fact holds for π_q, that is, $\pi_q(\Delta_\times)$ is a quantum coupling of $(\pi_q(\Delta_1), \pi_q(\Delta_2))$.

A.4 Semantics of Classical-Quantum Statements [25]

The operational and denotational semantics of our classical-quantum statements have been systematically studied in [25]. We here give a brief review and provide some useful propositions.

A configuration $\langle S, \sigma, \rho\rangle$ where $S \in \mathbf{Cmd} \cup \{E\}$, E is a special symbol to denote termination, $\sigma \in \Sigma$ and $\rho \in \mathcal{D}(qV)$ for some qV subsuming the quantum variables appeard in S. The operational semantics of programs in **Cmd** is defined as the smallest transition relation $\to$ on configurations given in Table 2.

Definition A.4 (Computation [25]). Let $c \in \mathbf{Cmd}$, $\langle \sigma, \rho\rangle \in \Xi(V)$ with $V \supseteq \mathrm{qVar}(c)$.

- A computation of c starting in $\langle \sigma, \rho\rangle$ if a (finite or infinite) maximal sequence of configurations $\langle c_i, \sigma_i, \rho_i\rangle$, such that

$$\langle c, \sigma, \rho\rangle \to \langle c_1, \sigma_1, \rho_1\rangle \to \langle c_2, \sigma_2, \rho_2\rangle \ \cdots$$

  and $\mathrm{tr}(\rho_i) > 0$ for all i.
- A computation of c is terminating in $\langle \sigma', \rho'\rangle$ if it is finite and the last configuration is $\langle E, \sigma', \rho'\rangle$; otherwise it is diverging.

Let $\to^n$ be the n-th composition of $\to$, and $\to^* \triangleq \bigcup_{n\geq 0} \to^n$.

Definition A.5 (Denotational semantics [25]). Let $c \in \mathbf{Cmd}$. The denotational semantics of c is a mapping

$$[\![c]\!] : \Xi(\supseteq \mathrm{qVar}(c)) \to \Xi(\supseteq \mathrm{qVar}(c))$$

such that for any $\langle \sigma, \rho\rangle \in \Xi(V)$ with $V \supseteq \mathrm{qVar}(c)$,

$$[\![c]\!](\langle \sigma, \rho\rangle) = \sum \left\{\!\!\left\{ \langle \sigma', \rho'\rangle : \langle c, \sigma, \rho\rangle \to^* \langle E, \sigma', \rho'\rangle \right\}\!\!\right\},$$

where $\{\!|\cdot|\!\}$ denotes the multi-set. Furthermore, we define $[\![c]\!](\Delta) = \sum_{i\in I} [\![c]\!](\langle \sigma_i, \rho_i\rangle)$ whenever $\Delta = \sum_{i\in I} \langle \sigma_i, \rho_i\rangle$ (linearity ensures this definition is well-defined).

Proposition A.6 ([25]). *For any $c \in \mathbf{Cmd}$ and $\Delta \in \Xi(V)$ with $V \supseteq qVar(c)$,*

(1) [trace-nonincreasing] $\mathrm{tr}([\![c]\!](\Delta)) \le \mathrm{tr}(\Delta)$, *and so* $[\![c]\!](\Delta) \in \Xi(V)$;
(2) [linearity] $[\![c]\!](\Delta) = \sum_i \lambda_i [\![c]\!](\Delta_i)$ *whenever* $\Delta = \sum_i \lambda_i \Delta_i$.

Proposition A.7 ([25]). *For any cq-state $\langle \sigma, \rho\rangle$ which makes the corresponding formula meaningful,*

(1) $[\![\mathsf{skip}]\!](\langle \sigma, \rho\rangle) = \langle \sigma, \rho\rangle$;
(2) $[\![x := e]\!](\langle \sigma, \rho\rangle) = \langle \sigma[[\![e]\!]_\sigma/x], \rho\rangle$;
(3) $[\![x \xleftarrow{\$} d]\!](\langle \sigma, \rho\rangle) = \sum_{v\in\lceil [\![d]\!]_\sigma \rceil} \langle \sigma[v/x], [\![d]\!]_\sigma(v)\cdot\rho\rangle$;
(4) $[\![x := \mathsf{meas}\, M[\overline{q}]]\!](\langle \sigma, \rho\rangle) = \sum_{i\in\mathcal{J}} \langle \sigma[i/x], ([\![M_i]\!]_\sigma)_{\overline{q}}\rho([\![M_i^\dagger]\!]_\sigma)_{\overline{q}}\rangle$ *where measurement* $M = \{M_i\}_{i\in\mathcal{J}}$;
(5) $[\![q := e]\!](\langle \sigma, \rho\rangle) = \langle \sigma, \sum_{i=0}^{d_q-1} |[\![e]\!]_\sigma\rangle_q\langle i|\rho|i\rangle_q\langle [\![e]\!]_\sigma|\rangle$;
(6) $[\![\overline{q} \mathrel{*}= U]\!](\langle \sigma, \rho\rangle) = \langle \sigma, ([\![U]\!]_\sigma)_{\overline{q}}\rho([\![U^\dagger]\!]_\sigma)_{\overline{q}}\rangle$;
(7) $[\![c_1; c_2]\!](\langle \sigma, \rho\rangle) = [\![c_2]\!]([\![c_1]\!](\langle \sigma, \rho\rangle))$;
(8) $[\![(c_1; c_2); c_3]\!] = [\![c_1; (c_2; c_3)]\!]$;
(9) $[\![\mathsf{if}\ b\ \mathsf{then}\ c_1\ \mathsf{else}\ c_2]\!](\langle \sigma, \rho\rangle) = [\![c_1]\!](\langle \sigma, \rho\rangle)$ *if* $\sigma \models b$, *and* $[\![c_2]\!](\langle \sigma, \rho\rangle)$ *otherwise;*
(10) $[\![\mathbf{while}]\!](\langle \sigma, \rho\rangle) = \bigsqcup_n([\![(\mathbf{while})^n]\!](\langle \sigma, \rho\rangle))$, *where* $\mathbf{while} \triangleq \mathsf{while}\ b\ \mathsf{do}\ c$, $(\mathbf{while})^0 \triangleq \mathbf{abort}$, *and for any* $n \ge 0$,

$$(\mathbf{while})^{n+1} \triangleq \mathsf{if}\ b\ \mathsf{then}\ c; (\mathbf{while}^n)\ \mathsf{else}\ \mathsf{skip}.$$

Proposition A.8 (c.f. [25]). *Let* $\mathbf{while} \triangleq \mathsf{while}\ b\ \mathsf{do}\ c$. *For any* $\Delta \in \Xi(qVar(c))$ *and* $n \ge 0$,

$$\Delta^{\neg b} + [\![(\mathbf{while})^n]\!]([\![c]\!](\Delta^b)) = [\![(\mathbf{while})^{n+1}]\!](\Delta).$$

where $\Delta^b \triangleq \sum_\sigma \mathbb{I}([\![b]\!]_\sigma, \mathbf{true})\cdot\langle \sigma, \Delta(\sigma)\rangle$ *and similar for* $\Delta^{\neg b}$. *In particular,*

$$\Delta^{\neg b} + [\![\mathbf{while}]\!]([\![c]\!](\Delta^b)) = [\![\mathbf{while}]\!](\Delta).$$

Furthermore, let $\Delta_1 = \Delta$ *and for any* $n \ge 1$, $\Delta_{n+1} = [\![c]\!](\Delta_n^b)$. *Then for any* $n \ge 1$,

$$[\![(\mathbf{while})^n]\!](\Delta) = \sum_{i=1}^{n} \Delta_i^{\neg b}, \quad \textit{in particular,}\ [\![\mathbf{while}]\!](\Delta) = \sum_{i\ge 1} \Delta_i^{\neg b}.$$

Proof. The first part is proved in [25]. For the rest part, we can easily prove

$$\forall\, n \ge 1, k \ge 1,\ [\![(\mathbf{while})^n]\!](\Delta_k) = \sum_{i=k}^{n+k-1} \Delta_i$$

by induction on n, k. Then set $k = 1$ and let $n \to \infty$, we obtain the last equation. The convergence is guaranteed by the fact that Ξ is an ω-complete partial order set (CPO) under $\sqsubseteq$[25]. □

Proposition A.9. *For any $c \in \mathbf{Cmd}$, there exists a mapping*

$$f : \Sigma(Var(c)) \times \Sigma(Var(c)) \to (\mathcal{D}(qVar(c)) \to \mathcal{D}(qVar(c)))$$

where $\Sigma(S) \triangleq S \to \mathcal{V}$ *such that*[3]*:*

(1) $\forall\, \sigma_{\mathrm{in}}, \sigma_{\mathrm{out}}, f(\sigma_{\mathrm{in}}, \sigma_{\mathrm{out}})$ *is a trace-nonincreasing quantum operation, i.e., completely-positive trace-nonincreasing linear maps.*
(2) $\forall\, \sigma_{\mathrm{in}},\ \sum_{\sigma_{\mathrm{out}}\in\Sigma(Var(c))} f(\sigma_{\mathrm{in}}, \sigma_{\mathrm{out}})$ *is a trace-nonincreasing quantum operation; it is a trace-preserving quantum operation if c is lossless, i.e., c is almost surely terminating.*
(3) For any input singleton cq-states $\langle \sigma, \rho\rangle \in \Xi(V)$ with $V \supseteq qVar(c)$, split $\sigma = (\sigma_{\mathrm{in}}; \sigma_{\mathrm{res}})$ such that $\sigma_{\mathrm{in}} \in \Sigma(Var(c))$ and $\sigma_{\mathrm{res}} \in \Sigma(Var - Var(c))$

$$[\![c]\!](\langle \sigma, \rho\rangle) = \sum_{\sigma_{\mathrm{out}}\in\Sigma(Var(c))} \langle \sigma_{\mathrm{out}}; \sigma_{\mathrm{res}}, f(\sigma_{\mathrm{in}}, \sigma_{\mathrm{out}}) \otimes \mathcal{I}_{V-qVar(c)}(\rho)\rangle.$$

[3]We set $\Sigma(\emptyset) = \{\mathsf{empmem}\}$ with single element empmem denotes no assignment of any classical variables; thus $\sigma = (\sigma; \mathsf{empmem}) = (\mathsf{empmem}; \sigma)$.

As a corollary of third statements, it is always possible to extend f to a larger sets of classical/quantum variable sets, i.e., for any $S \supseteq Var(c)$ and $V \supseteq qVar(c)$, there exists $f' : \Sigma(S) \times \Sigma(S) \to (\mathcal{D}(V) \to \mathcal{D}(V))$ that satisfies all 1-3 statements.

In particular, if c has no quantum initialization and quantum measurement, then for any σ_{in} and σ_{out}, $f(\sigma_{\text{in}}, \sigma_{\text{out}})$ is a trace-preserving quantum operation with a factor.

The proof is easy but tedious. In principle, it can be regarded as the generalization of Proposition 3.3.5 in [51].

Proof. We prove all the three statements by induction on the structure of c. For any c, we always set $f(\sigma_{\text{in}}, \sigma_{\text{out}}) = 0$ if $\sigma_{\text{in}}, \sigma_{\text{out}}$ are not consistent, i.e., they are different on some variables that are not modified by $\text{Var}(c)$.

- $c \equiv \mathsf{skip}$. Since $\text{Var}(c) = \text{qVar}(c) = \emptyset$, $\sigma_{\text{in}} = \sigma_{\text{out}} = \mathsf{empmem}$, set

$$f(\mathsf{empmem}, \mathsf{empmem}) = 1.$$

  Then first two are trivial and for the third one,

$$[\![\mathsf{skip}]\!](\langle \sigma, \rho\rangle) = \langle \sigma, \rho\rangle = \langle \mathsf{empmem}; \sigma, f(\mathsf{empmem}, \mathsf{empmem}) \otimes \mathcal{I}_V(\rho)\rangle$$

- $c \equiv x := e$. $\text{Var}(c) = \text{Var}(e) \cup \{x\}$, $\text{qVar}(c) = \emptyset$. For all $\sigma_{\text{in}}, \sigma_{\text{out}}$, set

$$f(\sigma_{\text{in}}, \sigma_{\text{out}}) = \mathbb{I}(\sigma_{\text{out}}, \sigma_{\text{in}}[[\![e]\!]_{\sigma_{\text{in}}}/x]).$$

  Then first two are trivial. For the third one,

$$\begin{aligned}
&\sum_{\sigma_{\text{out}} \in \Sigma(\text{Var}(c))} \langle \sigma_{\text{out}}; \sigma_{\text{res}}, f(\sigma_{\text{in}}, \sigma_{\text{out}}) \otimes \mathcal{I}_{V-\text{qVar}(c)}(\rho)\rangle \\
= &\sum_{\sigma_{\text{out}} \in \Sigma(\text{Var}(c))} \langle \sigma_{\text{out}}; \sigma_{\text{res}}, \mathbb{I}(\sigma_{\text{out}}, \sigma_{\text{in}}[[\![e]\!]_{\sigma_{\text{in}}}/x])\mathcal{I}_V(\rho)\rangle \\
= &\langle \sigma_{\text{in}}[[\![e]\!]_{\sigma_{\text{in}}}/x]; \sigma_{\text{res}}, \rho\rangle = \langle \sigma[[\![e]\!]_{\sigma_{\text{in}}}/x], \rho\rangle \\
= &[\![x := e]\!](\langle \sigma, \rho\rangle)
\end{aligned}$$

- $c \equiv x \xleftarrow{\$} d$. $\text{Var}(c) = \text{Var}(d) \cup \{x\}$, $\text{qVar}(c) = \emptyset$. For all $\sigma_{\text{in}}, \sigma_{\text{out}}$, set

$$f(\sigma_{\text{in}}, \sigma_{\text{out}}) = \mathbb{I}(\sigma_{\text{out}}, \sigma_{\text{in}}[[\![x]\!]_{\sigma_{\text{out}}}/x])[\![d]\!]_{\sigma_{\text{in}}}([\![x]\!]_{\sigma_{\text{out}}}).$$

  The first statement is trivial. For the second statement, since $[\![d]\!]_{\sigma_{\text{in}}}$ is a distribution, we have:

$$\sum_{\sigma_{\text{out}}} f(\sigma_{\text{in}}, \sigma_{\text{out}}) = \sum_{v \in \left\lceil [\![d]\!]_{\sigma_{\text{in}}} \right\rceil} [\![d]\!]_{\sigma_{\text{in}}}([\![x]\!]_{\sigma_{\text{out}}}) = \sum_{v \in \left\lceil [\![d]\!]_{\sigma_{\text{in}}} \right\rceil} [\![d]\!]_{\sigma_{\text{in}}}(v) = 1.$$

  For the third statement,

$$\begin{aligned}
&\sum_{\sigma_{\text{out}} \in \Sigma(\text{Var}(c))} \langle \sigma_{\text{out}}; \sigma_{\text{res}}, f(\sigma_{\text{in}}, \sigma_{\text{out}}) \otimes \mathcal{I}_{V-\text{qVar}(c)}(\rho)\rangle \\
= &\sum_{\sigma_{\text{out}} \in \Sigma(\text{Var}(c))} \langle \sigma_{\text{out}}; \sigma_{\text{res}}, \mathbb{I}(\sigma_{\text{out}}, \sigma_{\text{in}}[[\![x]\!]_{\sigma_{\text{out}}}/x])[\![d]\!]_{\sigma_{\text{in}}}([\![x]\!]_{\sigma_{\text{out}}})\mathcal{I}_V(\rho)\rangle \\
= &\sum_{v \in \left\lceil [\![d]\!]_{\sigma_{\text{in}}} \right\rceil} \langle \sigma_{\text{out}}[v/x]; \sigma_{\text{res}}, [\![d]\!]_{\sigma_{\text{in}}}(v) \cdot \rho\rangle \\
= &\sum_{v \in \left\lceil [\![d]\!]_{\sigma_{\text{in}}} \right\rceil} \langle \sigma[v/x], [\![d]\!]_{\sigma_{\text{in}}}(v) \cdot \rho\rangle \\
= &[\![x \xleftarrow{\$} d]\!](\langle \sigma, \rho\rangle)
\end{aligned}$$

- $c \equiv x := \mathsf{meas}\, M[\overline{q}]$ with measurement $M = \{M_i\}_{i \in \mathcal{J}}$. $\text{Var}(c) = \text{Var}(M) \cup \{x\}$, $\text{qVar}(c) = \overline{q}$. For all $\sigma_{\text{in}}, \sigma_{\text{out}}$, set

$$\forall\, \rho \in \mathcal{D}(\overline{q}),\ f(\sigma_{\text{in}}, \sigma_{\text{out}}) = \sum_{i \in \mathcal{J}} \mathbb{I}(\sigma_{\text{out}}, \sigma_{\text{in}}[i/x])[\![M_i]\!]_{\sigma_{\text{in}}} \rho [\![M_i^\dagger]\!]_{\sigma_{\text{in}}}.$$

  The first statement is true since $M_i(\cdot)M_i$ is a completely-positive trace-nonincreasing map. For the second statement, we have:

$$\forall\, \rho \in \mathcal{D}(\overline{q}),\ \sum_{\sigma_{\text{out}}} f(\sigma_{\text{in}}, \sigma_{\text{out}}) = \sum_{\sigma_{\text{out}}} \sum_{i \in \mathcal{J}} \mathbb{I}(\sigma_{\text{out}}, \sigma_{\text{in}}[i/x])[\![M_i]\!]_{\sigma_{\text{in}}} \rho [\![M_i^\dagger]\!]_{\sigma_{\text{in}}} = \sum_{i \in \mathcal{J}} [\![M_i]\!]_{\sigma_{\text{in}}} \rho [\![M_i^\dagger]\!]_{\sigma_{\text{in}}}.$$

which is a completely-positive trace-perserving (CPTP) map, i.e., trace-preserving quantum operation. For the third statement,

$$\begin{aligned}
&\sum_{\sigma_{\text{out}} \in \Sigma(\text{Var}(c))} \langle \sigma_{\text{out}}; \sigma_{\text{res}}, f(\sigma_{\text{in}}, \sigma_{\text{out}}) \otimes \mathcal{I}_{V-\text{qVar}(c)}(\rho) \rangle \\
= &\sum_{\sigma_{\text{out}} \in \Sigma(\text{Var}(c))} \langle \sigma_{\text{out}}; \sigma_{\text{res}}, \sum_{i \in \mathcal{J}} \mathbb{I}(\sigma_{\text{out}}, \sigma_{\text{in}}[i/x]) [\![M_i]\!]_{\sigma_{\text{in}}} \rho [\![M_i^\dagger]\!]_{\sigma_{\text{in}}} \rangle \\
= &\sum_{i \in \mathcal{J}} \langle \sigma_{\text{out}}[i/x]; \sigma_{\text{res}}, [\![M_i]\!]_{\sigma_{\text{in}}} \rho [\![M_i^\dagger]\!]_{\sigma_{\text{in}}} \rangle \\
= &\sum_{i \in \mathcal{J}} \langle \sigma[i/x], [\![M_i]\!]_{\sigma} \rho [\![M_i^\dagger]\!]_{\sigma} \rangle \\
= &[\![x := \text{meas } M[\overline{q}]]\!](\langle \sigma, \rho \rangle)
\end{aligned}$$

- $c \equiv q := e$. $\text{Var}(c) = \text{Var}(e)$, $\text{qVar}(c) = \{q\}$. For any $\sigma_{\text{in}},\ \sigma_{\text{out}}$, set

$$\forall\, \rho \in \mathcal{D}(q),\ f(\sigma_{\text{in}}, \sigma_{\text{out}}) = \mathbb{I}(\sigma_{\text{in}}, \sigma_{\text{out}}) \cdot \sum_{i=0}^{d_q - 1} |[\![e]\!]_{\sigma_{\text{in}},}\rangle_q \langle i|\rho|i\rangle_q \langle [\![e]\!]_{\sigma_{\text{in}},}|.$$

  The first two statements are trivial since initialization is a completely-positive trace-nonincreasing map. For the third statement, $\sigma = (\sigma_{\text{in}}; \sigma_{\text{res}})$,

$$\begin{aligned}
&\sum_{\sigma_{\text{out}} \in \Sigma(\text{Var}(c))} \langle \sigma_{\text{out}}; \sigma_{\text{res}}, f(\sigma_{\text{in}}, \sigma_{\text{out}}) \otimes \mathcal{I}_{V-q}(\rho) \rangle \\
= &\langle \sigma, \sum_{i=0}^{d_q - 1} |[\![e]\!]_{\sigma}\rangle_q \langle i|\rho|i\rangle_q \langle [\![e]\!]_{\sigma}| \rangle \\
= &[\![q := e]\!](\langle \sigma, \rho \rangle)
\end{aligned}$$

- $c \equiv \overline{q} \mathrel{*}= U$. $\text{Var}(c) = \text{Var}(U)$, $\text{qVar}(c) = \{\overline{q}\}$.For any $\sigma_{\text{in}},\ \sigma_{\text{out}}$, set

$$\forall\, \rho \in \mathcal{D}(q),\ f(\sigma_{\text{in}}, \sigma_{\text{out}}) = \mathbb{I}(\sigma_{\text{in}}, \sigma_{\text{out}}) \cdot ([\![U]\!]_{\sigma_{\text{in}}})_{\overline{q}} \rho ([\![U^\dagger]\!]_{\sigma_{\text{in}}})_{\overline{q}}.$$

  The first two statements are trivial since unitary transformation is a completely-positive trace-nonincreasing map. For the third statement, $\sigma = (\sigma_{\text{in}}; \sigma_{\text{res}})$,

$$\sum_{\sigma_{\text{out}} \in \Sigma(\text{Var}(c))} \langle \sigma_{\text{out}}; \sigma_{\text{res}}, f(\sigma_{\text{in}}, \sigma_{\text{out}}) \otimes \mathcal{I}_{V-q}(\rho) \rangle = \langle \sigma, ([\![U]\!]_{\sigma_{\text{in}}})_{\overline{q}} \rho ([\![U^\dagger]\!]_{\sigma_{\text{in}}})_{\overline{q}} \rangle = [\![\overline{q} \mathrel{*}= U]\!](\langle \sigma, \rho \rangle)$$

- $c \equiv c_1; c_2$. Suppose f_1 and f_2 is the mapping for c_1 and c_2 respectively. $\text{Var}(c) = \text{Var}(c_1) \cup \text{Var}(c_2)$, $\text{qVar}(c) = \text{qVar}(c_1) \cup \text{qVar}(c_2)$. For any $\sigma_{\text{in}}, \sigma_{\text{out}} \in \Sigma(\text{Var}(c))$, split $\sigma_{\text{in}} = (\sigma_{\text{in1}}; \sigma_{\text{res1}})$ and $\sigma_{\text{out}} = (\sigma_{\text{out2}}; \sigma_{\text{res2}})$, set

$$f(\sigma_{\text{in}}, \sigma_{\text{out}}) = \sum_{\sigma_{\text{out1}}} (f_2(\sigma_{\text{in2}}, \sigma_{\text{out2}}) \otimes \mathcal{I}_{\text{qVar}(c) - \text{qVar}(c_2)}) \circ (f_1(\sigma_{\text{in1}}, \sigma_{\text{out1}}) \otimes \mathcal{I}_{\text{qVar}(c) - \text{qVar}(c_1)}).$$

  where $(\sigma_{\text{in2}}; \sigma_{\text{res2}})$ is set to be $(\sigma_{\text{out1}}; \sigma_{\text{res1}})$, same for the following. For the first two statements, since the composition of completely-positive map is still completely positive, we only need to show they are trace-nonincreasing: for any $\rho \in \mathcal{D}(\text{qVar}(c))$:

$$\begin{aligned}
&\text{tr}\left(\sum_{\sigma_{\text{out}}} f(\sigma_{\text{in}}, \sigma_{\text{out}})(\rho) \right) \\
= &\text{tr}\Big[\sum_{\sigma_{\text{out2}}} \sum_{\sigma_{\text{out1}}} (f_2(\sigma_{\text{in2}}, \sigma_{\text{out2}}) \otimes \mathcal{I}_{\text{qVar}(c) - \text{qVar}(c_2)}) \circ (f_1(\sigma_{\text{in1}}, \sigma_{\text{out1}}) \otimes \mathcal{I}_{\text{qVar}(c) - \text{qVar}(c_1)})(\rho) \Big] \\
\le &\text{tr}\Big[\sum_{\sigma_{\text{out1}}} (f_1(\sigma_{\text{in1}}, \sigma_{\text{out1}}) \otimes \mathcal{I}_{\text{qVar}(c) - \text{qVar}(c_1)})(\rho) \Big] \\
\le &\text{tr}(\rho).
\end{aligned}$$

And if $c_1; c_2$ is lossless, then both c_1 and c_2 are trace-preserving, and above inequalities are all saturated. For the third statement, $\sigma = (\sigma_{\text{in}}; \sigma_{\text{res}}) = (\sigma_{\text{in}1}; \sigma_{\text{res}1}; \sigma_{\text{res}})$:

$$\begin{aligned}
&\sum_{\sigma_{\text{out}} \in \Sigma(\text{Var}(c))} \langle \sigma_{\text{out}}; \sigma_{\text{res}}, f(\sigma_{\text{in}}, \sigma_{\text{out}}) \otimes \mathcal{I}_{V-\text{qVar}(c)}(\rho)\rangle \\
&= \sum_{\sigma_{\text{out}2}} \langle \sigma_{\text{out}2}; \sigma_{\text{res}2}; \sigma_{\text{res}}, \sum_{\sigma_{\text{out}1}} (f_2(\sigma_{\text{in}2}, \sigma_{\text{out}2}) \otimes \mathcal{I}_{V-\text{qVar}(c_2)}) \circ (f_1(\sigma_{\text{in}1}, \sigma_{\text{out}1}) \otimes \mathcal{I}_{V-\text{qVar}(c_1)})(\rho)\rangle \\
&= \sum_{\sigma_{\text{out}1}} \sum_{\sigma_{\text{out}2}} \langle \sigma_{\text{out}2}; \sigma_{\text{res}2}; \sigma_{\text{res}}, (f_2(\sigma_{\text{in}2}, \sigma_{\text{out}2}) \otimes \mathcal{I}_{V-\text{qVar}(c_2)})\big((f_1(\sigma_{\text{in}1}, \sigma_{\text{out}1}) \otimes \mathcal{I}_{V-\text{qVar}(c_1)})(\rho)\big)\rangle \\
&= \sum_{\sigma_{\text{out}1}} [\![c_2]\!] \left[\langle \sigma_{\text{in}2}; \sigma_{\text{res}2}; \sigma_{\text{res}}, (f_1(\sigma_{\text{in}1}, \sigma_{\text{out}1}) \otimes \mathcal{I}_{V-\text{qVar}(c_1)})(\rho)\rangle\right] \\
&= [\![c_2]\!] \Big[\sum_{\sigma_{\text{out}1}} \langle \sigma_{\text{out}1}; \sigma_{\text{res}1}; \sigma_{\text{res}}, (f_1(\sigma_{\text{in}1}, \sigma_{\text{out}1}) \otimes \mathcal{I}_{V-\text{qVar}(c_1)})(\rho)\rangle \Big] \\
&= [\![c_2]\!] \left[[\![c_1]\!](\langle \sigma_{\text{in}1}; \sigma_{\text{res}1}; \sigma_{\text{res}}, \rho\rangle) \right] \\
&= [\![c_1; c_2]\!](\langle \sigma, \rho\rangle).
\end{aligned}$$

- $c \equiv$ if b then c_1 else c_2. Suppose f_1 and f_2 is the mapping for c_1 and c_2 respectively. $\text{Var}(c) = \text{Var}(c_1) \cup \text{Var}(c_2)$, $\text{qVar}(c) = \text{qVar}(c_1) \cup \text{qVar}(c_2)$. For any $\sigma_{\text{in}}, \sigma_{\text{out}} \in \Sigma(\text{Var}(c))$, split $\sigma_{\text{in}} = (\sigma_{\text{in}k}; \sigma_{\text{res}k})$ for $k = 1, 2$ and similar for σ_{out}, set

$$\begin{aligned}
f(\sigma_{\text{in}}, \sigma_{\text{out}}) = {} & \mathbb{I}([\![b]\!]_\sigma, \textbf{true}) \cdot (f_1(\sigma_{\text{in}1}, \sigma_{\text{out}1}) \otimes \mathcal{I}_{\text{qVar}(c)-\text{qVar}(c_1)}) \\
& + \mathbb{I}([\![b]\!]_\sigma, \textbf{false}) \cdot (f_2(\sigma_{\text{in}2}, \sigma_{\text{out}2}) \otimes \mathcal{I}_{\text{qVar}(c)-\text{qVar}(c_2)}).
\end{aligned}$$

where $(\sigma_{\text{in}2}; \sigma_{\text{res}2})$ is set to be $(\sigma_{\text{out}1}; \sigma_{\text{res}1})$, same for the following. For the first two statements, since the sum and composition of completely-positive map are still completely positive, we only need to show they are trace-nonincreasing: for any $\rho \in \mathcal{D}(\text{qVar}(c))$:

$$\begin{aligned}
&\text{tr}\left(\sum_{\sigma_{\text{out}}} f(\sigma_{\text{in}}, \sigma_{\text{out}})(\rho)\right) \\
&= \mathbb{I}([\![b]\!]_\sigma, \textbf{true}) \cdot \text{tr}\Big[\sum_{\sigma_{\text{out}1}} (f_1(\sigma_{\text{in}1}, \sigma_{\text{out}1}) \otimes \mathcal{I}_{\text{qVar}(c)-\text{qVar}(c_1)})(\rho) \Big] \\
&\quad + \mathbb{I}([\![b]\!]_\sigma, \textbf{false}) \cdot \text{tr}\Big[\sum_{\sigma_{\text{out}2}} (f_2(\sigma_{\text{in}2}, \sigma_{\text{out}2}) \otimes \mathcal{I}_{\text{qVar}(c)-\text{qVar}(c_2)})(\rho) \Big] \\
&\le \mathbb{I}([\![b]\!]_\sigma, \textbf{true})\text{tr}(\rho) + \mathbb{I}([\![b]\!]_\sigma, \textbf{false})\text{tr}(\rho) \\
&= \text{tr}(\rho).
\end{aligned}$$

And if if b then c_1 else c_2 is lossless, then both c_1 and c_2 are trace-preserving, and above inequality is saturated. For the third statement, $\sigma = (\sigma_{\text{in}1}; \sigma_{\text{res}1}) = (\sigma_{\text{in}2}; \sigma_{\text{res}2})$:

$$\begin{aligned}
&\sum_{\sigma_{\text{out}} \in \Sigma(\text{Var}(c))} \langle \sigma_{\text{out}}; \sigma_{\text{res}}, f(\sigma_{\text{in}}, \sigma_{\text{out}}) \otimes \mathcal{I}_{V-\text{qVar}(c)}(\rho)\rangle \\
&= \mathbb{I}([\![b]\!]_\sigma, \textbf{true}) \cdot \sum_{\sigma_{\text{out}1}} \langle \sigma_{\text{out}1}; \sigma_{\text{res}1}, (f_1(\sigma_{\text{in}1}, \sigma_{\text{out}1}) \otimes \mathcal{I}_{\text{qVar}(c)-\text{qVar}(c_1)})(\rho)\rangle \\
&\quad + \mathbb{I}([\![b]\!]_\sigma, \textbf{false}) \cdot \sum_{\sigma_{\text{out}2}} \langle \sigma_{\text{out}1}; \sigma_{\text{res}1}, (f_2(\sigma_{\text{in}2}, \sigma_{\text{out}2}) \otimes \mathcal{I}_{\text{qVar}(c)-\text{qVar}(c_2)})(\rho)\rangle \\
&= \mathbb{I}([\![b]\!]_\sigma, \textbf{true}) \cdot [\![c_1]\!](\langle \sigma, \rho\rangle) + \mathbb{I}([\![b]\!]_\sigma, \textbf{false}) \cdot [\![c_2]\!](\langle \sigma, \rho\rangle)
\end{aligned}$$

- $[\![\textbf{while}]\!](\langle \sigma, \rho\rangle) = \bigvee_n ([\![(\textbf{while})^n]\!](\langle \sigma, \rho\rangle)$, where **while** $\triangleq$ while b do c, $(\textbf{while})^0 \triangleq$ **abort**, and for any $n \ge 0$,

$$(\textbf{while})^{n+1} \triangleq \text{if } b \text{ then } c; (\textbf{while}^n) \text{ else skip}.$$

$c \equiv$ while b do c_1. $\text{Var}(c) = \text{Var}(b) \cup \text{Var}(c_1)$, $\text{qVar}(c) = \text{qVar}(c_1)$. Set following functions f_t, f_f:

$$\begin{aligned}
f_t(\sigma_{\text{in}}, \sigma_{\text{out}}) &= \mathbb{I}(\sigma_{\text{in}}, \sigma_{\text{out}})\mathbb{I}([\![b]\!]_{\sigma_{\text{in}}}, \textbf{true}) \cdot \mathcal{I}_{\text{qVar}(c_1)} \\
f_f(\sigma_{\text{in}}, \sigma_{\text{out}}) &= \mathbb{I}(\sigma_{\text{in}}, \sigma_{\text{out}})\mathbb{I}([\![b]\!]_{\sigma_{\text{in}}}, \textbf{false}) \cdot \mathcal{I}_{\text{qVar}(c_1)}
\end{aligned}$$

and f_{c_1} for the mapping of loop body c_1 extended to $\text{Var}(c)$. For f_1, f_2 with same Var, qVar, we define it's composition as:

$$(f_2 \circ f_1)(\sigma_{\text{in}}, \sigma_{\text{out}}) = \sum_{\sigma_{\text{mid}}} f_2(\sigma_{\text{mid}}, \sigma_{\text{out}}) \circ f_1(\sigma_{\text{in}}, \sigma_{\text{mid}}).$$

It is straightforward to show that composition is associative if it is well-defined. Define $f_k \triangleq \sum_{n=0}^{k} f_f \circ (f_c \circ f_t)^n$. We first show: for any $k \geq 0$,

$$\forall\, \rho \in \mathcal{D}(\mathrm{qVar}(c_1)),\ \mathrm{tr}\Big(\sum_{\sigma_{\mathrm{out}}} f_k(\sigma_{\mathrm{in}}, \sigma_{\mathrm{out}})(\rho)\Big) \leq \mathrm{tr}(\rho).$$

Base $k = 0$ is trivial. Suppose it holds for k. Consider $k = k + 1$: first $f_{k+1} = f_f + f_k \circ f_c \circ f_t$,

$$\begin{aligned}
&\mathrm{tr}\Big(\sum_{\sigma_{\mathrm{out}}} f_{k+1}(\sigma_{\mathrm{in}}, \sigma_{\mathrm{out}})(\rho)\Big) \\
&= \mathrm{tr}\Big(\sum_{\sigma_{\mathrm{out}}} f_f(\sigma_{\mathrm{in}}, \sigma_{\mathrm{out}})(\rho)\Big) + \mathrm{tr}\Big(\sum_{\sigma_{\mathrm{out}}} (f_k \circ f_c \circ f_t)(\sigma_{\mathrm{in}}, \sigma_{\mathrm{out}})(\rho)\Big) \\
&= \mathbb{I}(\llbracket b \rrbracket_{\sigma_{\mathrm{in}}}, \mathbf{false}) \cdot \mathrm{tr}(\rho) + \mathbb{I}(\llbracket b \rrbracket_{\sigma_{\mathrm{in}}}, \mathbf{true}) \cdot \mathrm{tr}\Big(\sum_{\sigma_{\mathrm{out}}} (f_k \circ f_c)(\sigma_{\mathrm{in}}, \sigma_{\mathrm{out}})(\rho)\Big) \\
&\leq \mathbb{I}(\llbracket b \rrbracket_{\sigma_{\mathrm{in}}}, \mathbf{false}) \cdot \mathrm{tr}(\rho) + \mathbb{I}(\llbracket b \rrbracket_{\sigma_{\mathrm{in}}}, \mathbf{true}) \cdot \mathrm{tr}(\rho) \\
&= \mathrm{tr}(\rho).
\end{aligned}$$

Since each term in f_k is completely-positive, so sequence f_k has limit, and we set $f = \lim_{k\to\infty} f_k$. Then the first and second statements hold; if c is lossless, then f_c is CPTP map when sum up the outputs, by definition the f is then a CPTP map. For the third statement, we first show that for all $n \geq 1$, (see Proposition A.8 for definitions and properties)

$$\sum_{\sigma_{\mathrm{out}}} \langle \sigma_{\mathrm{out}}; \sigma_{\mathrm{res}}, f_n(\sigma_{\mathrm{in}}, \sigma_{\mathrm{out}}) \otimes \mathcal{I}_{\mathrm{qVar}(c)-\mathrm{qVar}(c_1)})(\rho)\rangle = \llbracket (\mathbf{while})^n \rrbracket(\langle \sigma, \rho \rangle).$$

Base $n = 1$ is trivial. Suppose it holds for n. Consider $n = n + 1$. Since $(\mathbf{while})^{n+1} \triangleq$ if b then c_1; $(\mathbf{while})^n$ else skip,

$$\begin{aligned}
\llbracket (\mathbf{while})^n \rrbracket(\langle \sigma, \rho \rangle) &= \llbracket \text{if } b \text{ then } c_1; (\mathbf{while})^n \text{ else skip} \rrbracket(\langle \sigma, \rho \rangle) \\
&= \mathbb{I}(\llbracket b \rrbracket_{\sigma}, \mathbf{true}) \cdot \sum_{\sigma_{\mathrm{out}}} \langle \sigma_{\mathrm{out}}; \sigma_{\mathrm{res}}, ((f_n \circ f_{c_1})(\sigma_{\mathrm{in}}, \sigma_{\mathrm{out}}) \otimes \mathcal{I}_{\mathrm{qVar}(c)-\mathrm{qVar}(c_1)})(\rho)\rangle \\
&\quad + \mathbb{I}(\llbracket b \rrbracket_{\sigma}, \mathbf{false}) \cdot \langle \sigma, \mathcal{I}_{\mathrm{qVar}(c)}(\rho)\rangle \\
&= \sum_{\sigma_{\mathrm{out}}} \langle \sigma_{\mathrm{out}}; \sigma_{\mathrm{res}}, ((f_n \circ f_{c_1} \circ f_t)(\sigma_{\mathrm{in}}, \sigma_{\mathrm{out}}) \otimes \mathcal{I}_{\mathrm{qVar}(c)-\mathrm{qVar}(c_1)})(\rho)\rangle \\
&\quad + \sum_{\sigma_{\mathrm{out}}} \langle \sigma_{\mathrm{out}}; \sigma_{\mathrm{res}}, f_f(\rho)\rangle \\
&= \sum_{\sigma_{\mathrm{out}}} \langle \sigma_{\mathrm{out}}; \sigma_{\mathrm{res}}, ((f_f + f_n \circ f_{c_1} \circ f_t)(\sigma_{\mathrm{in}}, \sigma_{\mathrm{out}}) \otimes \mathcal{I}_{\mathrm{qVar}(c)-\mathrm{qVar}(c_1)})(\rho)\rangle \\
&= \sum_{\sigma_{\mathrm{out}}} \langle \sigma_{\mathrm{out}}; \sigma_{\mathrm{res}}, f_{n+1}(\sigma_{\mathrm{in}}, \sigma_{\mathrm{out}}) \otimes \mathcal{I}_{\mathrm{qVar}(c)-\mathrm{qVar}(c_1)})(\rho)\rangle.
\end{aligned}$$

Take $n \to \infty$, then we complete the proof.

The corollary is straightforwards if we define f' as: for all $\sigma'_{\mathrm{in}} = (\sigma_{\mathrm{in}}; \sigma_{\mathrm{res,in}})$ and $\sigma'_{\mathrm{out}} = (\sigma_{\mathrm{out}}; \sigma_{\mathrm{res,out}})$:

$$f'(\sigma'_{\mathrm{in}}, \sigma'_{\mathrm{out}}) = \mathbb{I}(\sigma_{\mathrm{res,in}}, \sigma_{\mathrm{res,out}}) \cdot f(\sigma_{\mathrm{in}}, \sigma_{\mathrm{out}}) \otimes \mathcal{I}_{V-\mathrm{qVar}(c)}.$$

□

A.5 Satisfaction Relation and Interpretation of Relational Assertions

We give the detailed definitions of satisfaction relation and interpretation in this subsection; they are the core of defining the validity of judgment.

Definition A.10 (Satisfaction relation). For any singleton product cq-state (product cq memory) $\langle \sigma_\times, \rho_\times \rangle$ and relational assertion Φ, the satisfaction relation $\langle \sigma_\times, \rho_\times \rangle \models \Phi$ is defined by induction on Φ:

$\langle \sigma_\times, \rho_\times \rangle \models p$ (classical atomic formulas)	iff	$\sigma_\times \models p$
$\langle \sigma_\times, \rho_\times \rangle \models (=_Q)$ (global quantum equality)	iff	$\rho_\times \models (=_Q)$, i.e., $\lceil \rho_\times \rceil \subseteq P_{\text{sym}}$[4]
$\langle \sigma_\times, \rho_\times \rangle \models \top$	iff	never
$\langle \sigma_\times, \rho_\times \rangle \models \bot$	iff	always
$\langle \sigma_\times, \rho_\times \rangle \models \Phi_1 \wedge \Phi_2$	iff	$\langle \sigma_\times, \rho_\times \rangle \models \Phi_1$ and $\langle \sigma_\times, \rho_\times \rangle \models \Phi_2$
$\langle \sigma_\times, \rho_\times \rangle \models \Phi_1 \vee \Phi_2$	iff	$\langle \sigma_\times, \rho_\times \rangle \models \Phi_1$ or $\langle \sigma_\times, \rho_\times \rangle \models \Phi_2$
$\langle \sigma_\times, \rho_\times \rangle \models \Phi_1 \implies \Phi_2$	iff	if $\langle \sigma_\times, \rho_\times \rangle \models \Phi_1$ then $\langle \sigma_\times, \rho_\times \rangle \models \Phi_2$
$\langle \sigma_\times, \rho_\times \rangle \models \forall x.\Phi$	iff	$\forall t.\langle \sigma_\times[t/x], \rho_\times \rangle \models \Phi$
$\langle \sigma_\times, \rho_\times \rangle \models \exists x.\Phi$	iff	$\exists t.\langle \sigma_\times[t/x], \rho_\times \rangle \models \Phi$

We then generalize it to general product cq-state $\Delta_\times, \Delta_\times \models \Phi$ if and only if

$$\forall \sigma_\times \in \lceil \Delta_\times \rceil,\ \langle \sigma_\times, \Delta_\times(\sigma_\times) \rangle \models \Phi.$$

Remark A.1. *Note that the satisfaction relation is first defined on singleton product cq-state and then generalize to all product cq-state, so for example, $\Delta_\times \models \Phi \implies \Psi$ implies if $\Delta_\times \models \Phi$ then $\Delta_\times \models \Psi$, but the converse is not true. This also happens in other connectives.*

Definition A.11 (cq-lifting of relational assertion). Suppose Φ is a relational assertion, we write the cq-lifting $\Delta_1 [\![\Phi]\!]^\# \Delta_2$ if there exists a cq-coupling $\Delta_\times$ of (Δ_1, Δ_2) such that $\Delta_\times \models \Phi$.

Definition A.12 (interpretation of Φ). The relational interpretation of Φ is the set over two singleton cq-states (cq memories) defined as follows:

$$[\![\Phi]\!] := \left\{ (\langle \sigma_1, \rho_1 \rangle, \langle \sigma_2, \rho_2 \rangle) \;\middle|\; \exists \text{ cq-coupling } \Delta_\times \text{ s.t. } \Delta_\times \models \Phi \right\}$$

Note that its classical memory of any possible cq-coupling $\Delta_\times$ is $(\sigma_1; \sigma_2)$.

Example A.13. Consider programs $c \equiv x \xleftarrow{\$}$ **Flip**, we have the valid judgment:

$$\models c \sim c : (=_Q) \Rightarrow (=_Q) \wedge x\langle 1 \rangle = x\langle 2 \rangle.$$

It asserts that, for any input pairs $\langle \sigma_1, \rho_1 \rangle$ and $\langle \sigma_2, \rho_2 \rangle$, if they satisfy $(=_Q)$ (this asserts that $\rho_1 = \rho_2$), then the outputs $\Delta_i(\sigma_i[k/x]) = \frac{1}{2}\rho_i$ (for $i = 1, 2$ and $k = 0, 1$) can lift to $(=_Q) \wedge x\langle 1 \rangle = x\langle 2 \rangle$, i.e., there exists identity coupling

$$\Delta_\times(x\langle 1 \rangle = k, x\langle 2 \rangle = k) = \frac{1}{2}\rho_1 \otimes \rho_2$$

that satisfies $(=_Q) \wedge x\langle 1 \rangle = x\langle 2 \rangle$.

A comparison of couplings in different paradigms is shown in Table 3.

Next, we present two technical propositions that relate the cq-lifting and interpretation.

Proposition A.14 (Equivalence of cq-lifting and interpretation for singleton cq-states). *For any relational assertion Φ, and singleton cq-state $\langle \sigma_1, \rho_2 \rangle, \langle \sigma_2, \rho_2 \rangle$, then*

$$\langle \sigma_1, \rho_1 \rangle [\![\Phi]\!]^\# \langle \sigma_2, \rho_2 \rangle \quad \textit{if and only if} \quad (\langle \sigma_1, \rho_1 \rangle, \langle \sigma_2, \rho_2 \rangle) \in [\![\Phi]\!].$$

Proposition A.15 (Equivalence of cq-lifting and interpretation). *For any relational assertion Φ, and cq-state Δ_1, Δ_2, the following statements are equivalent:*

(1) $\Delta_1 [\![\Phi]\!]^\# \Delta_2$;
(2) There exists a sequence of disjoint singleton pairs $(\langle \sigma_{i1}, \rho_{i1} \rangle, \langle \sigma_{i2}, \rho_{i2} \rangle) \in [\![\Phi]\!]$ such that $\forall i, j.\ i \neq j \rightarrow (\sigma_{i1}; \sigma_{i2}) \neq (\sigma_{j1}; \sigma_{j2})$ (this is what disjoint means) and $\Delta_1 = \sum_i \langle \sigma_{i1}, \rho_{i1} \rangle$ and $\Delta_2 = \sum_i \langle \sigma_{i2}, \rho_{i2} \rangle$.
(3) There exists a sequence of disjoint singleton states $\langle \sigma_{\times i}, \varrho_i \rangle \models \Phi$ such that $\forall i, j.\ (i \neq j \rightarrow \sigma_{\times i} \neq \sigma_{\times j})$ and $\Delta_1 = \sum_i \pi_1(\langle \sigma_{\times i}, \varrho_i \rangle)$ and $\Delta_2 = \sum_i \pi_2(\langle \sigma_{\times i}, \varrho_i \rangle)$.

[4] There are alternative ways to define equality. Here we use the symmetric space. As shown in [58], this implies $\text{tr}_{\text{qVar}\langle 1 \rangle}(\rho_\times) = \text{tr}_{\text{qVar}\langle 2 \rangle}(\rho_\times)$.

Classical	Product state: $\sigma_\times \in \Sigma_\times \triangleq \mathrm{Var}\langle 1\rangle \uplus \mathrm{Var}\langle 2\rangle \to \mathcal{V}$ $\sigma_\times$ is the "coupling" of (σ_1, σ_2) iff $\forall x \in \mathrm{Var}\langle 1\rangle,\ \sigma_\times(x) = \sigma_1\langle 1\rangle(x)$; $\forall x \in \mathrm{Var}\langle 2\rangle,\ \sigma_\times(x) = \sigma_2\langle 2\rangle(x)$ Assertion ϕ: first-order logic Satisfaction relation: $(\sigma_1, \sigma_2) \models \phi$ iff $\sigma_\times \models \phi$
Probabilistic	probability coupling: $\mu_\times \in \Sigma_\times \to [0, 1]$ $\mu_\times$ is a coupling of (μ_1, μ_2) iff $\forall \sigma_1,\ \sum_{\sigma_2} \mu_\times(\sigma_1; \sigma_2) = \mu_1(\sigma_1)$; $\forall \sigma_2,\ \sum_{\sigma_1} \mu_\times(\sigma_1; \sigma_2) = \mu_2(\sigma_2)$ Assertion ϕ: first-order logic Satisfaction relation: $\mu_1 [\![\phi]\!]^\# \mu_2$ iff $\exists$ a coupling $\mu_\times$ of (μ_1, μ_2) s.t. $\forall (\sigma_1; \sigma_2) \in \lceil \mu_\times \rceil,\ (\sigma_1, \sigma_2) \models \phi$
Quantum	quantum coupling: $\rho_\times \in \mathcal{D}(\mathrm{qVar}\langle 1\rangle \cup \mathrm{qVar}\langle 2\rangle)$ $\rho_\times$ is a coupling of (ρ_1, ρ_2) iff $\mathrm{tr}_{\mathrm{qVar}\langle 2\rangle}(\rho_\times) = \rho_1$; $\mathrm{tr}_{\mathrm{qVar}\langle 1\rangle}(\rho_\times) = \rho_2$ Assertion $(=_Q)$: quantum equality Satisfaction relation: $\rho_1 [\![(=_Q)]\!]^\# \rho_2$ iff $\exists$ a coupling $\rho_\times$ of (ρ_1, ρ_2) s.t. $\lceil \rho_\times \rceil \subseteq P_{\mathrm{sym}}$
Classical -Quantum	classical-quantum coupling: $\Delta_\times \in \Sigma_\times \to \mathcal{D}(\mathrm{qVar}\langle 1\rangle \cup \mathrm{qVar}\langle 2\rangle)$ $\Delta_\times$ is a coupling of (Δ_1, Δ_2) iff $\forall \sigma_1,\ \sum_{\sigma_2} \mathrm{tr}_{\mathrm{qVar}\langle 2\rangle}[\Delta_\times(\sigma_1; \sigma_2)] = \Delta_1(\sigma_1)$; $\forall \sigma_2,\ \sum_{\sigma_1} \mathrm{tr}_{\mathrm{qVar}\langle 1\rangle}[\Delta_\times(\sigma_1; \sigma_2)] = \Delta_2(\sigma_2)$ Assertion Φ: first-order logic on ϕ and $(=_Q)$ with bounded variables being classical Satisfaction relation: $\Delta_1 [\![\Phi]\!]^\# \Delta_2$ iff $\exists$ a coupling $\Delta_\times$ of (Δ_1, Δ_2) s.t. $\forall \sigma_\times \in \lceil \Delta_\times \rceil,\ \langle \sigma_\times, \Delta_\times(\sigma_\times)\rangle \models \Phi$

Table 3: Comparison of the couplings and satisfaction relations in different language paradigms.

Proof. $(1 \Rightarrow 2)$. Suppose $\sigma_\times$ is a witness of cq-lifting $\Delta_1 [\![\Phi]\!]^\# \Delta_2$, that is,

$$\forall (\sigma_1; \sigma_2) \in \lceil \Delta_\times \rceil \,.\, \langle \sigma_1; \sigma_2, \Delta_\times(\sigma_1; \sigma_2)\rangle \models \Phi.$$

Since $\Delta_\times$ has finite support, so we can list all different pairs $(\sigma_{i1}, \sigma_{i2}) \in \lceil \Delta_\times \rceil$, and define ρ_{i1}, ρ_{i2} as the marginal of $\Delta_\times(\sigma_{i1}; \sigma_{i2})$. Then, $(\langle \sigma_{i1}, \rho_{i1}\rangle, \langle \sigma_{i2}, \rho_{i2}\rangle) \in [\![\Phi]\!]$. And it is also trivial that:

$$\sum_i \langle \sigma_{i1}, \rho_{i1}\rangle = \sum_{(\sigma_1,\sigma_2) \in \lceil \Delta_\times \rceil} \langle \sigma_1, \mathrm{tr}_{\mathrm{qVar}_2}(\Delta_\times(\sigma_1; \sigma_2))\rangle = \sum_{\sigma_1} \langle \sigma_1, \sum_{\sigma_2} \mathrm{tr}_{\mathrm{qVar}_2}(\Delta_\times(\sigma_1; \sigma_2))\rangle = \sum_{\sigma_1} \langle \sigma_1, \Delta_1(\sigma_1)\rangle = \Delta_1$$

if we note that for any $(\sigma_1, \sigma_2) \notin \lceil \Delta_\times \rceil$, then $\Delta_\times(\sigma_1; \sigma_2) = 0$. Similar for $\Delta_2 = \sum_i \langle \sigma_{i2}, \rho_{i2}\rangle$.

$(2 \Rightarrow 1)$. Let us construct $\Delta_\times := \sum_i \langle \sigma_{i1}; \sigma_{i2}, \varrho_i\rangle$, where ϱ_i is a quantum coupling of (ρ_{i1}, ρ_{i2}) such that $\langle \sigma_{i1}; \sigma_{i2}, \varrho_i\rangle \models \Phi$ by definition of interpretation of Φ. It is straightforward to check that $\Delta_\times$ is a coupling of (Δ_1, Δ_2):

$$\pi_1(\Delta_\times) = \pi_1(\sum_i \langle \sigma_{i1}; \sigma_{i2}, \varrho_i\rangle) = \sum_i \langle \sigma_{i1}, \mathrm{tr}_{\mathrm{qVar}_2}(\varrho_i)\rangle = \sum_i \langle \sigma_{i1}, \rho_{i1}\rangle = \Delta_1.$$

On the other hand, for any $\sigma_\times \in \lceil \Delta_\times \rceil$, then there exists exactly one i such that $\Delta_\times = (\sigma_{i1}; \sigma_{i2})$, so $\Delta_\times(\sigma_\times) = \varrho_i$ and thus $\langle \Delta_\times, \varrho_i\rangle = \langle \sigma_{i1}; \sigma_{i2}, \varrho_i\rangle \models \Phi$ and then it implies $\Delta_\times$ is a witness of $\Delta_1 [\![\Phi]\!]^\# \Delta_2$.

$(2 \Leftrightarrow 3)$. Trivial. □

A.6 From judgments to probabilistic inequalities

A valid judgment gives a way to derive probabilistic (in)equality. For any classical relational assertion Φ, i.e., Φ contains no quantum equality $(=_Q)$, is fully determined by classical memories **CMem**, thus it can also be interpreted as a relation $[\![\Phi]\!]_c \subseteq \mathbf{CMem} \times \mathbf{CMem}$:

$$[\![\Phi]\!]_c \triangleq \{(\sigma_1, \sigma_2) \mid (\sigma_1; \sigma_2) \models \Phi\}.$$

We state the following fundamental lemma of cq-lifting with classical relational assertion as an extension of its counterpart in probabilistic setting:

Lemma A.16. *Let $E_1, E_2 \subseteq \mathbf{CMem}$. Let Φ be a classical relational assertion such that for every $(\sigma_1, \sigma_2) \in \mathbf{CMem} \times \mathbf{CMem}$, $(\sigma_1, \sigma_2) \in [\![\Phi]\!]_c$ implies $\sigma_1 \in E_1 \implies \sigma_2 \in E_2$. If $\Delta_1 [\![\Phi]\!]^\# \Delta_2$ then*

$$\Pr_{\Delta_1}[E_1] \leq \Pr_{\Delta_2}[E_2].$$

Proof. Since $\Delta_1 [\![\Phi]\!]^\# \Delta_2$, let $\Delta_\times$ be a witness of this cq-lifting, then we have: $\forall (\sigma_1; \sigma_2) \in \lceil \Delta_\times \rceil$, $\langle (\sigma_1; \sigma_2), \Delta_\times(\sigma_1; \sigma_2) \rangle \models \Phi$. Note that Φ is classical, so $\forall (\sigma_1; \sigma_2) \in \lceil \Delta_\times \rceil$, $(\sigma_1; \sigma_2) \models \Phi$, this leads to $\sigma_1 \in E_1 \implies \sigma_2 \in E_2$, or equivalently, using identity function, $\mathbb{I}(\sigma_1 \in E_1) \leq \mathbb{I}(\sigma_2 \in E_2)$.

On the other hand, let us unfold the probabilities of events E_1 and E_2 using the fact that $\Delta_\times$ is a coupling of (Δ_1, Δ_2):

$$\Pr_{\Delta_1}[E_1] = \sum_{\sigma_1 \in E_1} \mathrm{tr}[\Delta_1(\sigma_1)] = \sum_{\sigma_1 \in E_1} \mathrm{tr}\left[\sum_{\sigma_2} \mathrm{tr}_{\mathrm{qVar}\langle 2\rangle}[\Delta_\times(\sigma_1; \sigma_2)]\right] = \sum_{(\sigma_1;\sigma_2) \in \lceil \Delta_\times \rceil} \mathbb{I}(\sigma_1 \in E_1) \cdot [\Delta_\times(\sigma_1; \sigma_2)]$$

$$\Pr_{\Delta_2}[E_2] = \sum_{\sigma_2 \in E_2} \mathrm{tr}[\Delta_2(\sigma_2)] = \sum_{\sigma_2 \in E_2} \mathrm{tr}\left[\sum_{\sigma_1} \mathrm{tr}_{\mathrm{qVar}\langle 1\rangle}[\Delta_\times(\sigma_1; \sigma_2)]\right] = \sum_{(\sigma_1;\sigma_2) \in \lceil \Delta_\times \rceil} \mathbb{I}(\sigma_2 \in E_2) \cdot [\Delta_\times(\sigma_1; \sigma_2)]$$

which directly concludes $\Pr_{\Delta_1}[E_1] \leq \Pr_{\Delta_2}[E_2]$. □

Corollary A.17. *Let $E_1, E_2 \subseteq \mathbf{CMem}$. Let Φ be a classical relational assertion such that for every $(\sigma_1, \sigma_2) \in \mathbf{CMem} \times \mathbf{CMem}$, $(\sigma_1, \sigma_2) \in [\![\Phi]\!]_c$ implies $\sigma_1 \in E_1 \Leftrightarrow \sigma_2 \in E_2$. If $\Delta_1 [\![\Phi]\!]^\# \Delta_2$ then*

$$\Pr_{\Delta_1}[E_1] = \Pr_{\Delta_2}[E_2].$$

Based on it, we are able to bound probabilities of events via certain post-conditions. Essentially, an event determined by the classical memory is independent of its quantum memory, so it is safe to discard the quantum system; on the other hand, all the definitions of cq-state, cq-coupling, cq-lifting reduce to probabilistic setting if no quantum system is considered. Thus, it is not surprising that fundamental lemmas and corresponding lemmas used to derive probabilistic (in)equality in probabilistic setting still hold in our classical-quantum setting whenever relational assertions are classical (i.e., without quantum equality). We list some of the useful lemmas below: except the precondition Φ, all formulas are classical:

Lemma A.18. *If $\models c_1 \sim c_2 : \Phi \Rightarrow \phi_1\langle 1\rangle \implies \phi_2\langle 2\rangle$, then for every two classical-quantum memories Δ_1 and Δ_2 such that $(\Delta_1, \Delta_2) \in [\![\Phi]\!]$, we have:*

$$\Pr_{[\![c_1]\!](\Delta_1)}[[\![\phi_1]\!]] \leq \Pr_{[\![c_2]\!](\Delta_2)}[[\![\phi_2]\!]].$$

Lemma A.19. *If $\models c_1 \sim c_2 : \Phi \Rightarrow \phi_1\langle 1\rangle \implies (\phi_2\langle 2\rangle \vee F\langle 2\rangle)$, then for every two classical-quantum memories Δ_1 and Δ_2 such that $(\Delta_1, \Delta_2) \in [\![\Phi]\!]$, we have:*

$$\Pr_{[\![c_1]\!](\Delta_1)}[[\![\phi_1]\!]] - \Pr_{[\![c_2]\!](\Delta_2)}[[\![\phi_2]\!]] \leq \Pr_{[\![c_2]\!](\Delta_2)}[[\![F]\!]].$$

Lemma A.20. *If $\models c_1 \sim c_2 : \Phi \Rightarrow (\phi_1\langle 1\rangle \wedge F_1\langle 1\rangle) \Longleftrightarrow (\phi_2\langle 2\rangle \wedge F_2\langle 2\rangle)$, then for every two classical-quantum memories Δ_1 and Δ_2 such that $(\Delta_1, \Delta_2) \in [\![\Phi]\!]$, we have:*

$$\Pr_{[\![c_1]\!](\Delta_1)}[[\![\phi_1]\!]] - \Pr_{[\![c_2]\!](\Delta_2)}[[\![\phi_2]\!]] \leq \max(\Pr_{[\![c_1]\!](\Delta_1)}[[\![\neg F_1]\!]],\ \Pr_{[\![c_2]\!](\Delta_2)}[[\![\neg F_2]\!]]).$$

Lemma A.21. *If $\models c_1 \sim c_2 : \Phi \Rightarrow \bigwedge_{x \in \mathcal{X}} x\langle 1\rangle = x\langle 2\rangle$, then for every two classical-quantum memories Δ_1 and Δ_2 such that $(\Delta_1, \Delta_2) \in [\![\Phi]\!]$, and for every event ϕ that only depends on $\mathcal{X}$, we have:*

$$\Pr_{[\![c_1]\!](\Delta_1)}[[\![\phi]\!]] = \Pr_{[\![c_2]\!](\Delta_2)}[[\![\phi]\!]].$$

Lemma A.22. *If $\models c_1 \sim c_2 : \Phi \Rightarrow \neg F\langle 2\rangle \rightarrow \bigwedge_{x \in \mathcal{X}} x\langle 1\rangle = x\langle 2\rangle)$, then for every two classical-quantum memories Δ_1 and Δ_2 such that $(\Delta_1, \Delta_2) \in [\![\Phi]\!]$, and for every event ϕ that only depends on $\mathcal{X}$, we have:*

$$\Pr_{[\![c_1]\!](\Delta_1)}[[\![\phi]\!]] - \Pr_{[\![c_2]\!](\Delta_2)}[[\![\phi]\!]] \leq \Pr_{[\![c_2]\!](\Delta_2)}[[\![F]\!]].$$

Lemma A.23. *If $\models c_1 \sim c_2 : \Phi \Rightarrow (F\langle 1\rangle \Leftrightarrow F\langle 2\rangle) \wedge (\neg F\langle 2\rangle \rightarrow \bigwedge_{x \in \mathcal{X}} x\langle 1\rangle = x\langle 2\rangle)$, then for every two classical-quantum memories Δ_1 and Δ_2 such that $(\Delta_1, \Delta_2) \in [\![\Phi]\!]$, and for every event ϕ that only depends on $\mathcal{X}$, we have:*

$$\left|\Pr_{[\![c_1]\!](\Delta_1)}[[\![\phi]\!]] - \Pr_{[\![c_2]\!](\Delta_2)}[[\![\phi]\!]]\right| \leq \max(\Pr_{[\![c_1]\!](\Delta_1)}[[\![F]\!]],\ \Pr_{[\![c_2]\!](\Delta_2)}[[\![F]\!]]).$$

A.7 Soundness of Inference Rules

In this section, we gives the necessary materials and the proofs for the soundness theorem 4.7 for the inference rules displayed in Figures 4, 5, 6 and 3. We make the convention that event typically refers to classical event ([Case]) and expressions are also classical (this is consistent with our syntax, e.g., in [While] and [Cond], e is also used as the guard in commands, so it should be classical); so we do not add side-conditions that they are classical.

To deal with closed-under-mixture (CM) condition, we further define the weak-lifting and weak satisfaction relation, which is closed under mixture for any cq-states.

Definition A.24 (weak-lifting of relational assertion). We say a cq-coupling $\Delta_\times$ weakly satisfies a relational assertion Φ iff

- There exists decomposition $\Delta_\times = \sum_{i\in\mathcal{J}} \Delta_{\times i}$ with **finite/countably infinite** indexing set $\mathcal{J}$;
- $\forall\, i \in \mathcal{J},\ \Delta_{\times i} \models \Phi$.

And we use notation $[\![\Phi]\!]_{\mathrm{w}}$ to denote the set of cq-couplings that weakly satisfies Φ.

Proposition A.25. *We have the following relations for arbitrary cq-states $\Delta\langle 1\rangle, \Delta\langle 2\rangle$ and relational assertion Φ.*

(1) $\Delta\langle 1\rangle [\![\Phi]\!]^{\#} \Delta\langle 2\rangle$ *implies* $\Delta\langle 1\rangle [\![\Phi]\!]^{\#}_{\mathrm{w}} \Delta\langle 2\rangle$.

(2) $\Delta\langle 1\rangle [\![\Phi]\!]^{\#}_{\mathrm{w}} \Delta\langle 2\rangle$ *and* cm (Φ) *implies* $\Delta\langle 1\rangle [\![\Phi]\!]^{\#} \Delta\langle 2\rangle$.

(3) If $\Phi \implies \Psi$, *then* $[\![\Phi]\!]_{\mathrm{w}} \subseteq [\![\Psi]\!]_{\mathrm{w}}$.
$[\![\Phi \wedge \Psi]\!]_{\mathrm{w}} \subseteq [\![\Phi]\!]_{\mathrm{w}} \cap [\![\Psi]\!]_{\mathrm{w}}$.
$[\![\Phi \vee \Psi]\!]_{\mathrm{w}} \supseteq [\![\Phi]\!]_{\mathrm{w}} \cup [\![\Psi]\!]_{\mathrm{w}}$.

(4) Weak satisfaction is closed under mixture, i.e., for a finite or countably infinite index set $\mathcal{J}$, and any sequence $\{\Delta_{\times i}\}_{i\in\mathcal{J}}$ such that $\Delta_{\times i} \in [\![\Phi]\!]_{\mathrm{w}}$ and $\lambda_i \in [0,1]$ such that $\sum_{i\in\mathcal{J}} \lambda_i \mathrm{tr}(\Delta_{\times i}) \le 1$, then

$$\sum_{i\in\mathcal{J}} \lambda_i \Delta_{\times i} \in [\![\Phi]\!]_{\mathrm{w}}.$$

Proof. • Trivial.

- Let $\Delta_\times$ be a witness of $\Delta\langle 1\rangle [\![\Phi]\!]^{\#}_{\mathrm{w}} \Delta\langle 2\rangle$ and decompose $\Delta_\times = \sum_{i\in\mathcal{J}} \Delta_{\times i}$ with finite/countably infinite indexing set $\mathcal{J}$ such that $\forall\, i \in \mathcal{J},\ \Delta_{\times i} \models \Phi$. Since cm (Φ), by Definition 4.6 and set $\lambda_i = 1$, we obtain $\sum_{i\in\mathcal{J}} \Delta_{\times i} \models \Phi$, or equivalently, $\Delta_\times \models \Phi$. Thus, $\Delta\langle 1\rangle [\![\Phi]\!]^{\#} \Delta\langle 2\rangle$ with a witness $\Delta_\times$.
- Since for any $\Delta_\times \in [\![\Phi]\!]_{\mathrm{w}}$ with decomposition $\Delta_\times = \sum_{i\in\mathcal{J}} \Delta_{\times i}$, $\Delta_{\times i} \models \Phi$ so $\Delta_{\times i} \models \Psi$, and thus $\Delta_\times \in [\![\Psi]\!]_{\mathrm{w}}$. Similarly for $[\![\Phi \wedge \Psi]\!]_{\mathrm{w}} \subseteq [\![\Phi]\!]_{\mathrm{w}} \cap [\![\Psi]\!]_{\mathrm{w}}$ and $[\![\Phi \vee \Psi]\!]_{\mathrm{w}} \supseteq [\![\Phi]\!]_{\mathrm{w}} \cup [\![\Psi]\!]_{\mathrm{w}}$.
- The proof is based on the fact that Cartesian product of two countable sets is still countable. Suppose for each $\Delta_{\times i}$ there is a decomposition $\sum_{j\in\mathcal{J}_i} \Delta_{\times ij}$ such that $\Delta_{\times ij} \models \Phi$, so $\lambda_i \Delta_{\times ij} \models \Phi$. On the other hand,

$$\sum_i \lambda_i \Delta_{\times i} = \sum_i \sum_{j\in\mathcal{J}_i} \lambda_i \Delta_{\times ij}$$

with a countable set of index since $\mathcal{J}, \mathcal{J}_i$ are countable, this leads to the conclusion.

□

Definition A.26 (Weak Validity of Judgment). Judgment $\models^{\mathrm{w}} c_1 \sim c_2 : \Phi \Rightarrow \Psi$ is valid, if for any two inputs $\langle \sigma_1, \rho_1\rangle$ for c_1 and $\langle \sigma_2, \rho_2\rangle$ for c_2 such that $(\langle \sigma_1, \rho_1\rangle, \langle \sigma_2, \rho_2\rangle) \in [\![\Phi]\!]$, the outputs

$$[\![c_1]\!](\langle \sigma_1, \rho_1\rangle) [\![\Psi]\!]^{\#}_{\mathrm{w}} [\![c_2]\!](\langle \sigma_2, \rho_2\rangle).$$

Proposition A.27 (Alternative Definition of Weak Validity). *Judgment $\models^{\mathrm{w}} c_1 \sim c_2 : \Phi \Rightarrow \Psi$ is valid, iff for any two inputs Δ_1 for c_1 and Δ_2 for c_2 such that $\Delta_1 [\![\Phi]\!]^{\#}_{\mathrm{w}} \Delta_2$, the outputs*

$$[\![c_1]\!](\Delta_1) [\![\Psi]\!]^{\#}_{\mathrm{w}} [\![c_2]\!](\Delta_2).$$

Proof. The if part is trivial. We then prove the only if part by the fact that Cartesian product of two countable sets is still countable.

For any inputs Δ_1 for c_1 and Δ_2 for c_2 that $\Delta_1 [\![\Phi]\!]^{\#}_{\mathrm{w}} \Delta_2$, let $\Delta_\times$ be the witness with decomposition $\Delta_\times = \sum_{i\in\mathcal{J}} \Delta_{\times i}$ and each $\Delta_{\times i} \models \Phi$. For each $\Delta_{\times i}$, since its support is countable, we can list all $(\sigma_{ij1}, \sigma_{ij2}) \in \lceil \Delta_{\times i} \rceil$ with index $j \in \mathcal{J}_i$, and trivially

$$(\langle \sigma_{ij1}, \pi_1(\Delta_{\times i}(\sigma_{ij1}; \sigma_{ij2}))\rangle, \langle \sigma_{ij2}, \pi_2(\Delta_{\times i}(\sigma_{ij2}; \sigma_{ij2}))\rangle) \in [\![\Phi]\!].$$

By assumption,

$$[\![c_1]\!](\langle \sigma_{ij1}, \pi_1(\Delta_{\times i}(\sigma_{ij1}; \sigma_{ij2}))\rangle) [\![\Psi]\!]^{\#}_{\mathrm{w}} [\![c_2]\!](\langle \sigma_{ij2}, \pi_2(\Delta_{\times i}(\sigma_{ij2}; \sigma_{ij2}))\rangle)$$

and let $\Delta'_{\times ij}$ be the witness. So $\Delta'_{\times ij} \in [\![\Psi]\!]_{\mathrm{w}}$. Now we construct the coupling $\Delta' = \sum_{i\in\mathcal{J}} \sum_{j\in\mathcal{J}_i} \Delta'_{\times ij}$. We next show that $\Delta'_\times$ is a coupling of $([\![c_1]\!](\Delta_1), [\![c_2]\!](\Delta_2))$.

$$\begin{aligned}
\pi_1(\Delta'_\times) &= \sum_{i\in\mathcal{J}} \sum_{j\in\mathcal{J}_i} \pi_1(\Delta'_{\times ij}) = \sum_{i\in\mathcal{J}} \sum_{j\in\mathcal{J}_i} [\![c_1]\!](\langle \sigma_{ij1}, \pi_1(\Delta_{\times i}(\sigma_{ij1}; \sigma_{ij2}))\rangle) \\
&= \sum_{i\in\mathcal{J}} [\![c_1]\!] \left(\sum_{j\in\mathcal{J}_i} \pi_1(\langle \sigma_{ij1}; \sigma_{ij2}, \Delta_{\times i}(\sigma_{ij1}; \sigma_{ij2})\rangle) \right) \\
&= \sum_{i\in\mathcal{J}} [\![c_1]\!](\pi_1(\Delta_{\times i})) = [\![c_1]\!] \Big(\pi_1 \Big(\sum_{i\in\mathcal{J}} \Delta_{\times i} \Big) \Big) \\
&= [\![c_1]\!](\pi_1(\Delta_\times)) = [\![c_1]\!](\Delta_1)
\end{aligned}$$

$$\frac{}{\vdash \mathsf{skip} \sim \mathsf{skip} : \Psi \Rightarrow \Psi}\ [\textsc{Skip}] \qquad \frac{\vdash c_1 \sim c_2 : \Phi \Rightarrow \Theta \quad \vdash c_1' \sim c_2' : \Theta \Rightarrow \Psi}{\vdash c_1; c_1' \sim c_2; c_2' : \Phi \Rightarrow \Psi}\ [\textsc{Seq}] \qquad \frac{}{\vdash x_1 := e_1 \sim x_2 := e_2 : \Psi[e_1\langle 1\rangle/x_1\langle 1\rangle][e_2\langle 2\rangle/x_2\langle 2\rangle] \Rightarrow \Psi}\ [\textsc{Assn}]$$

$$\frac{\begin{array}{c}\forall \sigma_1, \sigma_2. [\![\Phi]\!]_{\sigma_1,\sigma_2} \implies \blacktriangleleft_{[\![\Theta]\!]_{\sigma_1,\sigma_2}} \langle [\![d_1]\!]_{\sigma_1} \,\&\, [\![d_2]\!]_{\sigma_2}\rangle \\ \Phi \implies \forall v_1 : T_1, v_2 : T_2. (v_1 [\![\Theta]\!]^{\#} v_2 \implies \Psi[v_1/x_1\langle 1\rangle][v_2/x_2\langle 2\rangle])\end{array}}{\vdash x_1 \xleftarrow{\$} d_1 \sim x_2 \xleftarrow{\$} d_2 : \Phi \Rightarrow \Psi}\ [\textsc{Rand}] \qquad \frac{\begin{array}{c}\Phi \implies e_1\langle 1\rangle = e_2\langle 2\rangle \\ \vdash c_1 \sim c_2 : \Phi \wedge e_1\langle 1\rangle \Rightarrow \Psi \qquad \vdash c_1' \sim c_2' : \Phi \wedge \neg e_1\langle 1\rangle \Rightarrow \Psi\end{array}}{\vdash \mathsf{if}\ e_1\ \mathsf{then}\ c_1\ \mathsf{else}\ c_1' \sim \mathsf{if}\ e_2\ \mathsf{then}\ c_2\ \mathsf{else}\ c_2' : \Phi \Rightarrow \Psi}\ [\textsc{Cond}]$$

$$\frac{\Theta \implies e_1\langle 1\rangle = e_2\langle 2\rangle \qquad \vdash c_1 \sim c_2 : \Theta \wedge e_1\langle 1\rangle \Rightarrow \Theta}{\vdash \mathsf{while}\ e_1\ \mathsf{do}\ c_1 \sim \mathsf{while}\ e_2\ \mathsf{do}\ c_2 : \Theta \Rightarrow \Theta \wedge \neg e_1\langle 1\rangle}\ [\textsc{While}] \qquad \frac{\Phi \models f_1\langle 1\rangle + f_2\langle 1\rangle = f\langle 2\rangle}{\vdash \overline{q} \mathrel{*}= \mathrm{Uc}[f_1]; \overline{q} \mathrel{*}= \mathrm{Uc}[f_2] \sim \overline{q} \mathrel{*}= \mathrm{Uc}[f] : \Phi \Rightarrow \Phi}\ [\textsc{QROMComp}]$$

$$\frac{\Psi \implies f\langle 1\rangle = f\langle 2\rangle}{\vdash \overline{q} \mathrel{*}= \mathrm{Uc}[f] \sim \overline{q} \mathrel{*}= \mathrm{Uc}[f] : \Psi \Rightarrow \Psi}\ [\textsc{QROM}] \qquad \frac{\begin{array}{c}\Phi \implies \Phi'[e_1\langle 1\rangle/\mathbf{args}_{Q_1}\langle 1\rangle][e_2\langle 2\rangle/\mathbf{args}_{Q_2}\langle 2\rangle] \\ \vdash \mathbf{body}_{Q_1(\overline{q}_1)} \sim \mathbf{body}_{Q_2(\overline{q}_2)} : \Phi' \Rightarrow \Psi[\mathbf{res}_{Q_1}\langle 1\rangle/x_1\langle 1\rangle][\mathbf{res}_{Q_2}\langle 2\rangle/x_2\langle 2\rangle]\end{array}}{\vdash x_1 \leftarrow Q(e_1, \overline{q}_1) \sim x_2 \leftarrow Q(e_2, \overline{q}_2) : \Phi \Rightarrow \Psi}\ [\textsc{QCall}]$$

Figure 4: Two-sided rules for cq-commands. Remark: [QROM] displayed here is slightly stronger than in Figure 2.

$$\frac{}{\vdash x_1 := e_1 \sim \mathsf{skip} : \Psi[e_1\langle 1\rangle/x_1\langle 1\rangle] \Rightarrow \Psi}\ [\textsc{Assn-L}] \qquad \frac{}{\vdash x_1 \xleftarrow{\$} d_1 \sim \mathsf{skip} : \forall v_1 \in \lceil d_1 \rceil . \Psi[v_1/x_1\langle 1\rangle] \Rightarrow \Psi}\ [\textsc{Rand-L}]$$

$$\frac{\vdash c_1 \sim \mathsf{skip} : \Phi \wedge e_1\langle 1\rangle \Rightarrow \Psi \qquad \vdash c_1' \sim \mathsf{skip} : \Phi \wedge \neg e_1\langle 1\rangle \Rightarrow \Psi}{\vdash \mathsf{if}\ e_1\ \mathsf{then}\ c_1\ \mathsf{else}\ c_1' \sim \mathsf{skip} : \Phi \Rightarrow \Psi}\ [\textsc{Cond-L}] \qquad \frac{\vdash c_1 \sim \mathsf{skip} : \Theta \wedge e_1\langle 1\rangle \Rightarrow \Theta \qquad \mathsf{ast}\ (\mathsf{while}\ e_1\ \mathsf{do}\ c_1)}{\vdash \mathsf{while}\ e_1\ \mathsf{do}\ c_1 \sim \mathsf{skip} : \Theta \Rightarrow \Theta \wedge \neg e_1\langle 1\rangle}\ [\textsc{While-L}]$$

$$\frac{\mathsf{cls}(\Psi)}{\vdash \overline{q}_1 \mathrel{*}= \mathrm{Uc}[f] \sim \mathsf{skip} : \Psi \Rightarrow \Psi}\ [\textsc{QROM-L}] \qquad \frac{\begin{array}{c}\Phi \implies \Phi'[e_1\langle 1\rangle/\mathbf{args}_{Q_1}\langle 1\rangle] \\ \vdash \mathbf{body}_{Q_1(\overline{q}_1)} \sim \mathsf{skip} : \Phi' \Rightarrow \Psi[\mathbf{res}_{Q_1}\langle 1\rangle/x_1\langle 1\rangle]\end{array}}{\vdash x_1 \leftarrow Q(e_1, \overline{q}_1) \sim \mathsf{skip} : \Phi \Rightarrow \Psi}\ [\textsc{QCall-L}]$$

Figure 5: One-side rules for cq-commands.

Similar for $\pi_2(\Delta_\times') = [\![c_2]\!](\Delta_2)$. Note that the index set of $\sum_{i\in\mathcal{J}} \sum_{j\in\mathcal{J}_i} \Delta'_{\times ij}$ is still countable since $\mathcal{J}, \mathcal{J}_i$ are countable, thus $\Delta_\times' \in [\![\Psi]\!]_{\mathrm{w}}$ by Proposition A.25(4). □

Proposition A.28. *We have the following facts:*

(1) $\models \mathsf{skip} \sim \mathsf{skip} : \Psi \Rightarrow \Psi$.
(2) If $\models c_1 \sim c_2 : \Phi \Rightarrow \Psi$ *and* $\Phi' \implies \Phi$ *and* $\Psi \implies \Psi'$, *then* $\models c_1 \sim c_2 : \Phi' \Rightarrow \Psi'$.
(3) If $\models^{\mathrm{W}} c_1 \sim c_2 : \Phi \Rightarrow \Psi$ *and* $\mathsf{cm}(\Psi)$, *then* $\models c_1 \sim c_2 : \Phi \Rightarrow \Psi$.
(4) If $\models c_1 \sim c_2 : \Phi \Rightarrow \Psi$, *then* $\models^{\mathrm{W}} c_1 \sim c_2 : \Phi \Rightarrow \Psi$.
(5) If $\models^{\mathrm{W}} c_1 \sim c_2 : \Phi \Rightarrow \Theta$ *and* $\models^{\mathrm{W}} c_1' \sim c_2' : \Theta \Rightarrow \Psi$, *then* $\models^{\mathrm{W}} c_1; c_1' \sim c_2; c_2' : \Phi \Rightarrow \Psi$.

Proof. (1) Trivial.
(2) It is trivial if we use the fact that, if $\Phi \implies \Psi$, then $\Delta \models \Phi$ implies $\Delta \models \Psi$.
(3) Trivial by Proposition A.25 (2).
(4) Trivial by Proposition A.25 (1).
(5) It is trivial if we adopt the alternative definition by Proposition A.27.

□

We here prove the soundness of proof systems with respect to weak validity. Formally,

Theorem A.29 (Soundness w.r.t. Weak Validity). $\vdash c_1 \sim c_2 : \Phi \Rightarrow \Psi$ *implies* $\models^{\mathrm{W}} c_1 \sim c_2 : \Phi \Rightarrow \Psi$.

Then, Theorem 4.7 is just a corollary by Proposition A.28 (3).

• [Skip] Trivial. □

• [Seq] Trivial by Proposition A.28 (5). □

$$\frac{}{\vdash c_1 \sim c_2 : \bot \Rightarrow \Phi}\ [\textsc{False}] \qquad \frac{\mathsf{ast}(c_1) \qquad \mathsf{ast}(c_2)}{\vdash c_1 \sim c_2 : \top \Rightarrow \top}\ [\textsc{True}] \qquad \frac{\vdash c_1 \sim c_2 : \Phi' \Rightarrow \Psi' \qquad \Phi \implies \Phi' \qquad \Psi' \implies \Psi}{\vdash c_1 \sim c_2 : \Phi \Rightarrow \Psi}\ [\textsc{Conseq}]$$

$$\frac{\vdash c_1 \sim c_2 : \Phi \wedge e \Rightarrow \Psi \qquad \vdash c_1 \sim c_2 : \Phi \wedge \neg e \Rightarrow \Psi}{\vdash c_1 \sim c_2 : \Phi \Rightarrow \Psi}\ [\textsc{Case}] \qquad \frac{\forall x : T.\, \vdash c_1 \sim c_2 : \Phi \Rightarrow \Psi \qquad x \notin \mathsf{free}(\Psi)}{\vdash c_1 \sim c_2 : \exists x : T.\, \Phi \Rightarrow \Psi}\ [\textsc{Exists}]$$

$$\frac{\vdash c_1 \sim c : \Phi_1 \Rightarrow \Psi_1 \qquad \vdash c \sim c_2 : \Phi_2 \Rightarrow \Psi_2 \qquad \mathsf{cls}(\Phi_1) \quad \mathsf{cls}(\Phi_2) \quad \mathsf{cls}(\Psi_1) \quad \mathsf{cls}(\Psi_2)}{\vdash c_1 \sim c_2 : \Phi_1 \circ \Phi_2 \Rightarrow \Psi_1 \circ \Psi_2}\ [\textsc{PTrans}] \qquad \frac{\vdash c_1 \sim c_2 : \Phi \Rightarrow \Psi \qquad \vdash c_1 \sim c_2 : \Phi' \Rightarrow \Psi'}{\vdash c_1 \sim c_2 : \Phi \vee \Phi' \Rightarrow \Psi \vee \Psi'}\ [\textsc{Disj}]$$

$$\vdash c \equiv c\ [\textsc{Refl}] \qquad \Phi \vdash \mathsf{while}\ e\ \mathsf{do}\ c \equiv \mathsf{while}\ e \wedge e'\ \mathsf{do}\ c; \mathsf{while}\ e\ \mathsf{do}\ c\ [\textsc{While-Split}]$$

$$\frac{\models c_1 \sim c_2 : \Phi \Rightarrow \Psi}{\vdash c_1 \sim c_2 : \Phi \Rightarrow \Psi}\ [\textsc{Weak}] \qquad \frac{\vdash c_1 \sim c_2 : \Phi \Rightarrow \Psi \qquad \Phi \models c_1 \equiv c_1' \qquad \Phi \models c_2 \equiv c_2'}{\vdash c_1' \sim c_2' : \Phi \Rightarrow \Psi}\ [\textsc{Struct}]$$

$$\frac{\vdash c_1 \sim c_2 : \Phi \Rightarrow \Psi \qquad \mathsf{cls}(\Theta) \qquad \mathsf{free}(\Theta) \cap (\mathrm{MV}(c_1)\langle 1\rangle \cup \mathrm{MV}(c_2)\langle 2\rangle) = \emptyset}{\vdash c_1 \sim c_2 : \Phi \wedge \Theta \Rightarrow \Psi \wedge \Theta}\ [\textsc{PFrame}] \qquad \frac{\vdash c_1 \sim c_2 : \Phi \Rightarrow \Psi \qquad \mathsf{prob}(c_1) \qquad \mathsf{prob}(c_2) \qquad \mathsf{free}(\Theta) \cap (\mathrm{MV}(c_1)\langle 1\rangle \cup \mathrm{MV}(c_2)\langle 2\rangle) = \emptyset}{\vdash c_1 \sim c_2 : \Phi \wedge \Theta \Rightarrow \Psi \wedge \Theta}\ [\textsc{QFrame}]$$

Figure 6: Structural rules. $\mathsf{eqcmem}(S) \triangleq \bigwedge_{x \in S} x\langle 1\rangle = x\langle 2\rangle$ **asserts the equivalence of classical variables in set** S**, and** $\mathsf{eqmem}_{c_1,c_2,\ldots} \triangleq \mathsf{eqcmem}(\bigcup_i \mathbf{Var}(c_i)) \wedge (=_Q)$**. Then,** $\models c \equiv c'$ **is the abbreviation of** $\models c \sim c' : \mathsf{eqmem}_{c,c'} \Rightarrow \mathsf{eqmem}_{c,c'}$**.**

• [Assn] By Proposition A.28 (4), it is sufficient to show

$$\models x_1 := e_1 \sim x_2 := e_2 : \Psi[e_1\langle 1\rangle/x_1\langle 1\rangle][e_2\langle 2\rangle/x_2\langle 2\rangle] \Rightarrow \Psi.$$

For any inputs $\langle \sigma_1, \rho_1\rangle$ and $\langle \sigma_2, \rho_2\rangle$ satisfies $\Psi[e_1\langle 1\rangle/x_1\langle 1\rangle][e_2\langle 2\rangle/x_2\langle 2\rangle]$, there must exists a coupling $\Delta_\times = \langle \sigma_1; \sigma_2, \varrho\rangle$ such that ϱ is a q-coupling of (ρ_1, ρ_2) and $\langle \sigma_1; \sigma_2, \varrho\rangle \models \Psi[e_1\langle 1\rangle/x_1\langle 1\rangle][e_2\langle 2\rangle/x_2\langle 2\rangle]$.

By the denotational semantics of assignment, two outputs are

$$[\![x_1 := e_1]\!](\langle \sigma_1, \rho_1\rangle) = (\langle \sigma_1[[\![e_1]\!]_{\sigma_1}/x_1], \rho_1\rangle)$$
$$[\![x_2 := e_2]\!](\langle \sigma_2, \rho_2\rangle) = (\langle \sigma_2[[\![e_2]\!]_{\sigma_2}/x_2], \rho_2\rangle).$$

We define

$$\Delta_\times' = \langle \sigma_1[[\![e_1]\!]_{\sigma_1}/x_1]; \sigma_2[[\![e_2]\!]_{\sigma_2}/x_2], \varrho\rangle$$

which is a cq-coupling of outputs and satisfies Ψ. Note that the ϱ is preserved, so

$$\langle \sigma_1; \sigma_2, \varrho\rangle \models \Psi[e_1\langle 1\rangle/x_1\langle 1\rangle][e_2\langle 2\rangle/x_2\langle 2\rangle] \quad \text{iff} \quad \langle \sigma_1[[\![e_1]\!]_{\sigma_1}/x_1]; \sigma_2[[\![e_2]\!]_{\sigma_2}/x_2], \varrho\rangle \models \Psi$$

□

• [Rand] By Proposition A.28 (4), it is sufficient to show

$$\models x_1 \xleftarrow{\$} d_1 \sim x_2 \xleftarrow{\$} d_2 : \Phi \Rightarrow \Psi.$$

For any inputs $\langle \sigma_1, \rho_1\rangle$ and $\langle \sigma_2, \rho_2\rangle$ satisfies Φ, let $\langle \sigma_1; \sigma_2, \varrho\rangle$ be the witness of lifting, i.e., ϱ is a q-coupling of (ρ_1, ρ_2) and $\langle \sigma_1; \sigma_2, \varrho\rangle \models \Phi$.

By first premise we have $\blacktriangleleft_{[\![\Theta]\!]_{\sigma_1,\sigma_2}} \langle [\![d_1]\!]_{\sigma_1}\ \&\ [\![d_2]\!]_{\sigma_2}\rangle$, that is, there exists a witness $[\![d_\times]\!]_{\sigma_1,\sigma_2}$ of the lifting. So for any $(v_1, v_2) \in T_1 \times T_2$, $[\![d_\times]\!]_{\sigma_1,\sigma_2}(v_1, v_2) \geq 0$ implies $(v_1, v_2) \in \left\lceil [\![\Theta]\!]_{\sigma_1,\sigma_2} \right\rceil$. By Proposition A.7, we have two outputs

$$[\![x_1 \xleftarrow{\$} d_1]\!](\langle \sigma_1, \rho_1\rangle) = \sum_{v_1 \in \left\lceil [\![d_1]\!]_{\sigma_1} \right\rceil} \langle \sigma_1[v_1/x_1], [\![d_1]\!]_{\sigma_1}(v_1) \cdot \rho_1\rangle$$
$$[\![x_2 \xleftarrow{\$} d_2]\!](\langle \sigma_2, \rho_2\rangle) = \sum_{v_2 \in \left\lceil [\![d_2]\!]_{\sigma_2} \right\rceil} \langle \sigma_2[v_2/x_2], [\![d_2]\!]_{\sigma_2}(v_2) \cdot \rho_2\rangle$$

and construct

$$\begin{aligned}\Delta_\times &= \sum_{(v_1,v_2)\in T_1\times T_2} \langle \sigma_1[v_1/x_1\langle 1\rangle]; \sigma_2[v_2/x_2\langle 2\rangle], [\![d_\times]\!]_{\sigma_1,\sigma_2}(v_1,v_2)\cdot\varrho\rangle \\ &= \sum_{(v_1,v_2)\in\left\lceil [\![\Theta]\!]_{\sigma_1,\sigma_2}\right\rceil} \langle \sigma_1[v_1/x_1\langle 1\rangle]; \sigma_2[v_2/x_2\langle 2\rangle], [\![d_\times]\!]_{\sigma_1,\sigma_2}(v_1,v_2)\cdot\varrho\rangle\end{aligned}$$

which is the coupling of two outputs,

$$\begin{aligned}\pi_1(\Delta_\times) &= \sum_{(v_1,v_2)\in\left\lceil [\![d_2]\!]_{\sigma_1}\right\rceil\times\left\lceil [\![d_2]\!]_{\sigma_2}\right\rceil} \langle \sigma_1[v_1/x_1], [\![d_\times]\!]_{\sigma_1,\sigma_2}(v_1,v_2)\cdot \mathrm{tr}_{\mathrm{qVar}_2}(\varrho)\rangle \\ &= \sum_{v_1\in\left\lceil [\![d_2]\!]_{\sigma_1}\right\rceil} \langle \sigma_1[v_1/x_1], \sum_{v_2\in\left\lceil [\![d_2]\!]_{\sigma_2}\right\rceil} [\![d_\times]\!]_{\sigma_1,\sigma_2}(v_1,v_2)\cdot\rho_1\rangle \\ &= \sum_{v_1\in\left\lceil [\![d_2]\!]_{\sigma_1}\right\rceil} \langle \sigma_1[v_1/x_1], [\![d_1]\!]_{\sigma_1}(v_1)\cdot\rho_1\rangle\end{aligned}$$

since $[\![d_\times]\!]_{\sigma_1,\sigma_2}$ is a coupling of $([\![d_1]\!]_{\sigma_1}, [\![d_2]\!]_{\sigma_2})$. Next, by leveraging the second part of the premise $\Phi \implies (\forall v_1 : T_1, v_2 : T_2.(v_1[\![\Theta]\!]^\# v_2 \implies \Psi[v_1/x_1\langle 1\rangle][v_2/x_2\langle 2\rangle]))$ and the assumption $\langle\sigma_1;\sigma_2,\varrho\rangle \models \Phi$, we know that $\langle\sigma_1;\sigma_2,\varrho\rangle \models \forall v_1 : T_1, v_2 : T_2.(v_1[\![\Theta]\!]^\# v_2 \implies \Psi[v_1/x_1\langle 1\rangle][v_2/x_2\langle 2\rangle])$, that is, for any $(v_1,v_2)\in\left\lceil [\![\Theta]\!]_{\sigma_1,\sigma_2}\right\rceil$, then

$$\langle\sigma_1;\sigma_2,\varrho\rangle \models \Psi[v_1/x_1\langle 1\rangle][v_2/x_2\langle 2\rangle],$$

which leads to $\langle\sigma_1[v_1/x_1];\sigma_2[v_2/x_2],\varrho\rangle \models \Psi$. Note that if $(v_1,v_2)\neq(v_1',v_2')$, then

$$(\sigma_1[v_1/x_1];\sigma_2[v_2/x_2]) \neq (\sigma_1[v_1'/x_1];\sigma_2[v_2'/x_2]),$$

so they are disjoint, and then by Proposition A.15, $\Delta_\times \models \Psi$. □

• [Cond] For any inputs $\langle\sigma_1,\rho_1\rangle$ and $\langle\sigma_2,\rho_2\rangle$ satisfies Φ, there are two cases of $[\![e_1\langle 1\rangle]\!]_{\sigma_1}$. If $[\![e_1\langle 1\rangle]\!]_{\sigma_1} = \textbf{true}$, by premise $\Phi \implies e_1\langle 1\rangle = e_2\langle 2\rangle$, then $[\![e_2\langle 2\rangle]\!]_{\sigma_2} = \textbf{true}$. According to Proposition A.7, we have

$$\begin{aligned}&[\![\textsf{if } e_1 \textsf{ then } c_1 \textsf{ else } c_1']\!](\langle\sigma_1,\rho_1\rangle) = [\![c_1]\!](\langle\sigma_1,\rho_1\rangle)\\ &[\![\textsf{if } e_2 \textsf{ then } c_2 \textsf{ else } c_2']\!](\langle\sigma_2,\rho_2\rangle) = [\![c_2]\!](\langle\sigma_2,\rho_2\rangle).\end{aligned}$$

By premise and induction hypothesis $\models^{\mathrm{W}} c_1 \sim c_2 : \Phi \wedge e_1\langle 1\rangle \Rightarrow \Psi$, so the outputs

$$[\![\textsf{if } e_1 \textsf{ then } c_1 \textsf{ else } c_1']\!](\langle\sigma_1,\rho_1\rangle)[\![\Psi]\!]^\#_{\mathrm{W}}[\![\textsf{if } e_2 \textsf{ then } c_2 \textsf{ else } c_2']\!](\langle\sigma_2,\rho_2\rangle).$$

Similar for the case $[\![e_1\langle 1\rangle]\!]_{\sigma_1} = \textbf{false}$. □

• [While] We use Proposition A.27 as the definition. First by induction hypothesis, from the second premise we have

$$\models^{\mathrm{W}} c_1 \sim c_2 : \Theta \wedge e_1\langle 1\rangle \Rightarrow \Theta.$$

For any inputs Δ_1 and Δ_2 satisfies Θ, let $\Delta_{\times 1}$ be the witness of weak lifting $\Delta_1[\![\Theta]\!]^\#_{\mathrm{W}}\Delta_2$. According to Proposition A.8, we first define for any $n\geq 1$, $\Delta_{1,n+1} \triangleq [\![c_1]\!](\Delta_{1,n}^{e_1})$ with $\Delta_{1,1}\langle 1\rangle \triangleq \Delta_1$, and similarly define $\Delta_{2,n}$. Then we first have

$$[\![\textsf{while } e_1 \textsf{ do } c_1]\!](\Delta_1) = \sum_{n\geq 1}\Delta_{1,n}^{\neg e_1}, \quad [\![\textsf{while } e_2 \textsf{ do } c_2]\!](\Delta_2) = \sum_{n\geq 1}\Delta_{2,n}^{\neg e_2}.$$

Next, we prove the following statement:

$$\bullet \forall\, n\geq 1,\ \exists \Delta_{\times n} \text{ which is a witness of weak lifting } \Delta_{1,n}[\![\Theta]\!]^\#_{\mathrm{W}}\Delta_{2,n}.$$

The base case is trivial. Suppose the statement holds for n and consider the case of $n+1$. Let $\Delta_{\times n}$ be the witness of weak lifting $\Delta_{1,n}[\![\Theta]\!]^{\#}_{\mathrm{w}}\Delta_{2,n}$ with decomposition $\Delta_{\times n} = \sum_{i\in\mathcal{J}_n}\Delta_{\times ni}$ such that $\Delta_{\times ni} \models \Theta$. First, we observe:

$$\begin{aligned}
\Delta_{\times n}^{e_1\langle 1\rangle} &= \sum_{\sigma_1,\sigma_2} \mathbb{I}([\![e_1]\!]_{\sigma_1}, \mathbf{true})\langle\sigma_1;\sigma_2, \Delta_{\times n}(\sigma_1;\sigma_2)\rangle \\
&= \sum_{\sigma_1,\sigma_2} \mathbb{I}([\![e_1]\!]_{\sigma_1}, \mathbf{true})\langle\sigma_1;\sigma_2, \sum_{i\in\mathcal{J}_n}\Delta_{\times ni}(\sigma_1;\sigma_2)\rangle \\
&= \sum_{i\in\mathcal{J}_n}\sum_{\sigma_1,\sigma_2} \mathbb{I}([\![e_1]\!]_{\sigma_1}, \mathbf{true})\langle\sigma_1;\sigma_2, \Delta_{\times ni}(\sigma_1;\sigma_2)\rangle \\
&= \sum_{i\in\mathcal{J}_n} \Delta_{\times ni}^{e_1\langle 1\rangle}
\end{aligned}$$

and $\Delta_{\times ni}^{e_1\langle 1\rangle} \models \Theta \wedge e_1\langle 1\rangle$, so $\Delta_{\times n}^{e_1\langle 1\rangle} \in [\![\Theta \wedge e_1\langle 1\rangle]\!]_{\mathrm{w}}$ by Proposition A.25(4). Similarly $\Delta_{\times n}^{\neg e_1\langle 1\rangle} \in [\![\Theta \wedge \neg e_1\langle 1\rangle]\!]_{\mathrm{w}}$. On the other hand, by premise $\Theta \implies e_1\langle 1\rangle = e_2\langle 2\rangle$, we show that $\Delta_{\times n}^{e_1\langle 1\rangle}$ is a coupling of $(\Delta_{1,n}^{e_1}, \Delta_{2,n}^{e_2})$ as follows:

$$\begin{aligned}
\pi_1(\Delta_{\times n}^{e_1\langle 1\rangle}) &= \pi_1\Big(\sum_{\sigma_1,\sigma_2} \mathbb{I}([\![e_1]\!]_{\sigma_1}, \mathbf{true})\langle\sigma_1;\sigma_2, \Delta_{\times n}(\sigma_1;\sigma_2)\rangle\Big) \\
&= \Big(\sum_{\sigma_1} \mathbb{I}([\![e_1]\!]_{\sigma_1}, \mathbf{true})\langle\sigma_1, \sum_{\sigma_2}\mathrm{tr}_{\mathrm{qVar}\langle 2\rangle}[\Delta_{\times n}(\sigma_1;\sigma_2)]\rangle\Big) \\
&= \Big(\sum_{\sigma_1} \mathbb{I}([\![e_1]\!]_{\sigma_1}, \mathbf{true})\langle\sigma_1, \Delta_{1,n}(\sigma_1)\rangle\Big) \\
&= \Delta_{1,n}^{e_1}
\end{aligned}$$

and similar for $\pi_2(\Delta_{\times n}^{e_1\langle 1\rangle}) = \Delta_{2,n}^{e_2}$ by use the premise which implies $[\![e_1]\!]_{\sigma_1} = [\![e_2]\!]_{\sigma_2}$ if $(\sigma_1;\sigma_2) \in \lceil\Delta_{\times n}\rceil$. We can further show that $\Delta_{\times n}^{\neg e_1\langle 1\rangle}$ is a coupling of $(\Delta_{1,n}^{\neg e_1}, \Delta_{2,n}^{\neg e_2})$. Then by premise $\models^{\mathrm{w}} c_1 \sim c_2 : \Theta \wedge e_1\langle 1\rangle \Rightarrow \Theta$ we know that $[\![c_1]\!](\Delta_{1,n}^{e_1})[\![\Theta]\!]^{\#}_{\mathrm{w}}[\![c_2]\!](\Delta_{2,n}^{e_2})$, or equivalently, $\Delta_{1,n+1}[\![\Theta]\!]^{\#}\Delta_{2,n+1}$, and set the witness as $\Delta_{\times n+1}$.

Now we set $\Delta_{\times} \triangleq \sum_{n\geq 1}\Delta_{\times n}^{\neg e_1\langle 1\rangle}$. As proved above, for any $n \geq 1$, $\Delta_{\times n}^{\neg e_1\langle 1\rangle}$ is a cq-coupling of $(\Delta_{1,n}^{\neg e_1}, \Delta_{2,n}^{\neg e_2})$ and $\Delta_{\times n}^{\neg e_1\langle 1\rangle} \in [\![\Theta \wedge \neg e_1\langle 1\rangle]\!]_{\mathrm{w}}$. Thus, by Proposition A.25(4), $\Delta_{\times} \in [\![\Theta \wedge \neg e_1\langle 1\rangle]\!]_{\mathrm{w}}$. And it is also straightforward to see that $\Delta_{\times}$ is also a coupling of

$$([\![\mathsf{while}\ e_1\ \mathsf{do}\ c_1]\!](\Delta_1), [\![\mathsf{while}\ e_2\ \mathsf{do}\ c_2]\!](\Delta_2)).$$

□

• [QROM] We directly prove $\models \overline{q} \ast\!= \mathrm{Uc}[f] \sim \overline{q} \ast\!= \mathrm{Uc}[f] : \Psi \Rightarrow \Psi$. For any inputs $\langle\sigma_1,\rho_1\rangle$ and $\langle\sigma_2,\rho_2\rangle$ satisfies Ψ with witness $\langle\sigma_1;\sigma_2,\varrho\rangle$, by premise $f_1 = f_2$, so $\mathrm{Uc}[f_1] = \mathrm{Uc}[f_2] = U$, and thus the outputs are $\langle\sigma_1, U\rho_1U^\dagger\rangle$ and $\langle\sigma_2, U\rho_2U^\dagger\rangle$. Note that, $\varrho' \triangleq (U\otimes U)\varrho(U\otimes U)^\dagger$ is a quantum coupling of $(U\rho_1U^\dagger, U\rho_2U^\dagger)$, and $\varrho \models (=_Q)$ iff $\varrho' \models (=_Q)$, thus $(\langle\sigma_1, U\rho_1U^\dagger\rangle, \langle\sigma_2, U\rho_2U^\dagger\rangle) \models \Psi$ with witness $\langle\sigma_1;\sigma_2,\varrho'\rangle$. □

• [QROMComp] Realize that $\mathrm{Uc}[f_2]\cdot\mathrm{Uc}[f_1] = \mathrm{Uc}[f_1 + f_2]$: for any $|x,y\rangle$, $\mathrm{Uc}[f_1]|x,y\rangle = |x, y\oplus f_1(x)\rangle$, and then

$$\mathrm{Uc}[f_2]\cdot\mathrm{Uc}[f_1]|x,y\rangle = \mathrm{Uc}[f_2]|x, y\oplus f_1(x)\rangle = |x, y\oplus f_1(x)\oplus f_2(x)\rangle = |x, y\oplus(f_1+f_2)(x)\rangle = \mathrm{Uc}[f_1+f_2]|x,y\rangle.$$

Then we can directly use [QROM] to conclude.

• [QCall] Decompose: $x \leftarrow Q(e,\overline{q}) \equiv \mathbf{args}_Q := e;\ \mathbf{body}_{Q(\overline{q})};\ x := \mathbf{res}_Q$. By [Assn] and the premise, we have following:

$$\begin{aligned}
&\models^{\mathrm{w}} \mathbf{args}_Q := e \sim \mathbf{args}_Q := e : \Phi'[e_1\langle 1\rangle/\mathbf{args}_{Q_1}\langle 1\rangle][e_2\langle 2\rangle/\mathbf{args}_{Q_2}\langle 2\rangle] \Rightarrow \Phi' \\
&\models^{\mathrm{w}} \mathbf{body}_{Q_1(\overline{q}_1)} \sim \mathbf{body}_{Q_2(\overline{q}_2)} : \Phi' \Rightarrow \Psi[\mathbf{res}_{Q_1}\langle 1\rangle/x_1\langle 1\rangle][\mathbf{res}_{Q_2}\langle 2\rangle/x_2\langle 2\rangle] \\
&\models^{\mathrm{w}} x := \mathbf{res}_Q \sim x := \mathbf{res}_Q : \Psi[\mathbf{res}_{Q_1}\langle 1\rangle/x_1\langle 1\rangle][\mathbf{res}_{Q_2}\langle 2\rangle/x_2\langle 2\rangle] \Rightarrow \Psi
\end{aligned}$$

Repeatly using [Seq], we obtain $\models^{\mathrm{w}} x_1 \leftarrow Q(e_1,\overline{q}_1) \sim x_2 \leftarrow Q(e_2,\overline{q}_2) : \Phi \Rightarrow \Omega$ as desired.

• [False]. Trivial since the precondition is unsatisfiable. □

• [True]. Since both c_1 and c_2 are lossless, so for any inputs that has a coupling, then outputs have the same trace, so there also exist trivial coupling. Therefore, $\models c_1 \sim c_2 : \top \Rightarrow \top$. □

• [Conseq]. It is trivial by Proposition A.25 (3).

• [Case]. For any inputs $\langle\sigma_1,\rho_1\rangle$ and $\langle\sigma_2,\rho_2\rangle$ satisfies Φ, there are two cases of $[\![e]\!]_{\sigma_1;\sigma_2}$. If $[\![e]\!]_{\sigma_1;\sigma_2}$ = **true**, then $\models^{\mathrm{w}} c_1 \sim c_2 : \Phi \wedge e \Rightarrow \Psi$ follows by first premise and induction hypothesis; similar for the case that $[\![e]\!]_{\sigma_1;\sigma_2}$ = **false**. □

• [Exists]. For any inputs pair $(\langle\sigma_1,\rho_1\rangle,\langle\sigma_2,\rho_2\rangle) \models \exists x:T.\,\Phi$, thus there exists $v:T$ such that $(\langle\sigma_1[v/x],\rho_1\rangle,\langle\sigma_2[v/x],\rho_2\rangle) \models \Phi$, which is equivalent to $(\langle\sigma_1,\rho_1\rangle,\langle\sigma_2,\rho_2\rangle) \models \Phi[v/x]$. By premise and induction hypothesis $\models^{\mathrm{w}} c_1 \sim c_2 : \Phi[v/x] \Rightarrow \Psi[v/x]$, and by the second premise, $\Psi[v/x] = \Psi$, thus

$$[\![c_1]\!](\langle\sigma_1,\rho_1\rangle)[\![\Psi]\!]^{\#}_{\mathrm{w}}[\![c_2]\!](\langle\sigma_2,\rho_2\rangle).$$

□

• [Disj]. For any $\langle\sigma_1,\rho_1\rangle$ and $\langle\sigma_2,\rho_2\rangle$ satisfy $\Phi\vee\Phi'$, since they are singleton, so

$$(\langle\sigma_1,\rho_1\rangle,\langle\sigma_2,\rho_2\rangle) \in [\![\Phi]\!] \qquad \text{or} \qquad (\langle\sigma_1,\rho_1\rangle,\langle\sigma_2,\rho_2\rangle) \in [\![\Phi']\!].$$

Then by premises and induction hypothesis, we have

$$[\![c_1]\!](\langle\sigma_1,\rho_1\rangle)[\![\Psi]\!]^{\#}_{\mathrm{w}}[\![c_2]\!](\langle\sigma_2,\rho_2\rangle) \quad \text{or} \quad [\![c_1]\!](\langle\sigma_1,\rho_1\rangle)[\![\Psi']\!]^{\#}_{\mathrm{w}}[\![c_2]\!](\langle\sigma_2,\rho_2\rangle),$$

which leads to $[\![c_1]\!](\langle\sigma_1,\rho_1\rangle)[\![\Psi\vee\Psi']\!]^{\#}_{\mathrm{w}}[\![c_2]\!](\langle\sigma_2,\rho_2\rangle)$ and complete the proof. □

• [PTrans]. Since all assertions are classical so also cm, it directly reduce to probabilistic setting by using tensor product as trivial quantum couplings. □

• [Weak]. Trivial by Proposition A.28 (4).

• [Refl]. We directly prove that $\models c \equiv c$. Let $S = \mathrm{Var}(c)$.

For any $\langle\sigma_1,\rho_1\rangle$ and $\langle\sigma_2,\rho_2\rangle$ satisfy $(=_Q)\wedge \mathsf{eqcmem}\,(S)$, then we must have $\sigma_1 = (\sigma_S;\sigma_{\mathrm{res}1})$ and $\sigma_2 = (\sigma_S;\sigma_{\mathrm{res}2})$ and $\rho_1=\rho_2=\rho$. Let $f:\Sigma(S)\times\Sigma(S)\to(\mathcal{D}(\mathrm{qVar})\to\mathcal{D}(\mathrm{qVar}))$ be the function of c according to Proposition A.9. Now, for $i=1,2$, we have

$$[\![c]\!](\langle\sigma_i,\rho\rangle) = \sum_{\sigma'_S}\langle\sigma'_S;\sigma_{\mathrm{res}i}, f(\sigma_S,\sigma'_S)(\rho)\rangle.$$

Construct coupling

$$\Delta_\times = \sum_{\sigma'_S}\langle\sigma'_S;\sigma_{\mathrm{res}1};\sigma'_S;\sigma_{\mathrm{res}2}, \frac{f(\sigma_S,\sigma'_S)(\rho)\otimes f(\sigma_S,\sigma'_S)(\rho)}{\mathrm{tr}(f(\sigma_S,\sigma'_S)(\rho))}\rangle$$

which trivially satisfies $(=_Q)\wedge\mathsf{eqcmem}\,(S)$. □

• [While-Split]. Trivial.

Lemma A.30. *For any c,c', let $S = Var(c)\cup Var(c')$ and $\overline{q} = qVar(c)\cup qVar(c')$ and $f_c, f_{c'} : \Sigma(S)\times\Sigma(S)\to(\mathcal{D}(\overline{q})\to\mathcal{D}(\overline{q}))$ be the extended function of c and c' according to Proposition A.9. Then*

$$\models c\equiv c' \quad \textit{iff} \quad f_c = f_{c'}.$$

Proof. The if part is trivial since for the same inputs, the outputs then are equal and note the fact that for equal quantum states there exist couplings lie in symmetric subspace. For the only if part, let us choose the same input $\langle\sigma,\rho\rangle$ for both c and c'. The outputs are

$$[\![c]\!](\langle\sigma,\rho\rangle) = \sum_{\sigma'}\langle\sigma', f_c(\sigma,\sigma')(\rho)\rangle$$
$$[\![c']\!](\langle\sigma,\rho\rangle) = \sum_{\sigma'}\langle\sigma', f_{c'}(\sigma,\sigma')(\rho)\rangle.$$

Since they can be lifted by $(=_Q)\wedge\mathsf{eqcmem}\,(S)$, let

$$\Delta_\times = \sum_{\sigma'}\langle\sigma';\sigma',\varrho_{\sigma'}\rangle$$

be the witness. Then for any σ', $\varrho_{\sigma'}|_{\mathrm{qVar}\langle1\rangle} = f_c(\sigma,\sigma')(\rho) = \varrho_{\sigma'}|_{\mathrm{qVar}\langle2\rangle} = f_{c'}(\sigma,\sigma')(\rho)$. Since σ and ρ are also arbitrary, so $f_c = f_{c'}$ as we desired. □

• [Struct]. For any $\langle\sigma_1,\rho_1\rangle$ and $\langle\sigma_2,\rho_2\rangle$ satisfy Φ, by the second and the third premises, we know that $f_{c_1}=f_{c_1'}$ and $f_{c_2}=f_{c_2'}$ by Lemma A.30 (may extend to larger set; but this is not important). So we have:

$$\begin{aligned}[\![c_1]\!](\langle\sigma_1,\rho_1\rangle) &= \sum_{\sigma_{\text{out1}}} \langle\sigma_{\text{out1}};\sigma_{\text{res1}}, f_{c_1}(\sigma_1,\sigma_{\text{out1}})(\rho_1)\rangle\\ &= \sum_{\sigma_{\text{out1}}} \langle\sigma_{\text{out1}};\sigma_{\text{res1}}, f_{c_1'}(\sigma_1,\sigma_{\text{out1}})(\rho_1)\rangle\\ &= [\![c_1']\!](\langle\sigma_1,\rho_1\rangle).\end{aligned}$$

Similarly $[\![c_2]\!](\langle\sigma_2,\rho_2\rangle)=[\![c_2']\!](\langle\sigma_2,\rho_2\rangle)$. Thus, the outputs

$$[\![c_1]\!](\langle\sigma_1,\rho_1\rangle)[\![\Psi]\!]^{\#}_{\text{w}}[\![c_2]\!](\langle\sigma_2,\rho_2\rangle).$$

• [Assn-L]. Similar to [Assn].

• [Rand-L]. Similar to [Rand].

• [Cond-L]. Similar to [Cond].

• [While-L]. Similar to [While].

• [QROM-L]. Since unitary transformation does not change classical memory and $\mathsf{cls}(\Psi)$, so $\models \overline{q}_1 *= \mathrm{Uc}[f] \sim \mathsf{skip} : \Psi \Rightarrow \Psi$.

• [QROM-R]. Similar to [QROM-L].

• [QCall-L]. Similar to [QCall].

• [PFrame] For any inputs pair $(\langle\sigma_1,\rho_1\rangle,\langle\sigma_2,\rho_2\rangle) \models \Phi\wedge\Theta$, note that the witness must be a singleton cq-state, thus we can split into: $(\langle\sigma_1,\rho_1\rangle,\langle\sigma_2,\rho_2\rangle) \models \Phi$ and $(\langle\sigma_1,\rho_1\rangle,\langle\sigma_2,\rho_2\rangle) \models \Theta$. By first premise, $[\![c_1]\!](\langle\sigma_1,\rho_1\rangle)[\![\Psi]\!]^{\#}_{\text{w}}[\![c_2]\!](\langle\sigma_2,\rho_2\rangle)$ and let $\Delta_\times$ be the witness with decomposition $\Delta_\times=\sum_i\Delta_{\times i}$ such that $\Delta_{\times i}\models\Psi$. It is sufficient to show $\Delta_{\times i}\models\Theta$, or equivalently, for any $\sigma_{\times i}\in\lceil\Delta_{\times i}\rceil$, $\langle\sigma_{\times i},\Delta_{\times i}(\sigma_{\times i})\rangle\models\Theta$. For any classical variable $x\notin \mathrm{MV}(c_1)_1\cup\mathrm{MV}(c_2)_2$, $[\![x]\!]_{\sigma_{\times i}}=[\![x]\!]_{\sigma_1,\sigma_2}$, and by premises $\mathsf{cls}(\Theta)$ and $\mathsf{free}(\Theta)\cap(\mathrm{MV}(c_1)\langle1\rangle\cup\mathrm{MV}(c_2\langle2\rangle)$, so satisfaction of Θ only depends on variables that are not changed, thus $[\![\Theta]\!]_{\sigma_{\times i}}=[\![\Theta]\!]_{\sigma_1,\sigma_2}$. Since $(\langle\sigma_1,\rho_1\rangle,\langle\sigma_2,\rho_2\rangle)\models\Theta$, so $\langle\sigma_{\times i},\Delta_{\times i}(\sigma_{\times i})\rangle\models\Theta$.

• [QFrame] For any inputs pair $(\langle\sigma_1,\rho_1\rangle,\langle\sigma_2,\rho_2\rangle)$, let us fix the classical parts $(\sigma_1;\sigma_2)$. Then $(\langle\sigma_1,\rho_1\rangle,\langle\sigma_2,\rho_2\rangle)\models\Phi$ or not only depends on the atomic proposition $(=_Q)$, so there are three cases:

- $(\langle\sigma_1,\rho_1\rangle,\langle\sigma_2,\rho_2\rangle)\models\Phi$ iff $(=_Q)$.
  (Argument 1) We choose arbitrary pure state $|\psi\rangle$ and set $\rho_1=\rho_2=|\psi\rangle\langle\psi|$. So such inputs satisfies Φ, and by premise, $[\![c_1]\!](\langle\sigma_1,|\psi\rangle\rangle)[\![\Psi]\!]^{\#}_{\text{w}}[\![c_2]\!](\langle\sigma_2,|\psi\rangle\rangle)$ and let $\Delta_\times$ be a witness with decomposition $\Delta_\times=\sum_i\Delta_{\times i}$ such that $\Delta_{\times i}\models\Psi$. For any $\sigma_{\times i}=(\sigma_1';\sigma_2')\in\lceil\Delta_{\times i}\rceil$, $\Delta_{\times i}(\sigma_{\times i})=\lambda_{i\sigma_{\times i}}|\psi\rangle|\psi\rangle$ since the coupling of pure state is unique. Note that $\langle\sigma_{\times i},\Delta_{\times i}(\sigma_{\times i})\rangle\models\Psi$, and $(=_Q)$, so $[\![\Psi]\!]_{\sigma_{\times i}}$ is true if $(=_Q)$.
  (Argument 2) Now, let back to the choose general quantum states ρ_1 and ρ_2 such that $(\langle\sigma_1,\rho_1\rangle,\langle\sigma_2,\rho_2\rangle)\models\Phi\wedge\Theta$. We must have $\rho_1=\rho_2$; further, $[\![\Theta]\!]_{\sigma_1,\sigma_2}$ is true if $(=_Q)$. We directly construct the coupling of outputs $\Delta'\triangleq\sum_i\sum_{\sigma_{\times i}\in\lceil\Delta_{\times i}\rceil}\langle\sigma_{\times i},\lambda_{i\sigma_{\times i}}\rho_1\otimes\rho_2\rangle$. It is straightforward to show that it's a coupling of two outputs. Further, for each $\langle\sigma_{\times i},\lambda_{i\sigma_{\times i}}\rho_1\otimes\rho_2\rangle$, $(=_Q)$ and so $[\![\Psi]\!]_{\sigma_{\times i}}$; and since $\mathsf{free}(\Theta)\cap(\mathrm{MV}(c_1)\langle1\rangle\cup\mathrm{MV}(c_2)\langle2\rangle)=\emptyset$, so $[\![\Theta]\!]_{\sigma_{\times i}}=[\![\Theta]\!]_{\sigma_1,\sigma_2}$ which is also true since $(=_Q)$, so $\langle\sigma_{\times i},\lambda_{i\sigma_{\times i}}\rho_1\otimes\rho_2\rangle\models\Psi\wedge\Theta$, which leads to $\Delta'\in[\![\Psi\wedge\Theta]\!]_{\text{w}}$.
- $(\langle\sigma_1,\rho_1\rangle,\langle\sigma_2,\rho_2\rangle)\models\Phi$ iff $\neg(=_Q)$. Similar to the first case.
- $(\langle\sigma_1,\rho_1\rangle,\langle\sigma_2,\rho_2\rangle)\models\Phi$ no matter $(=_Q)$ or $\neg(=_Q)$. Following the similar Argument 1 (by choosing equal/different pure states as inputs), we can find the sequence of two supports $S_{i\mathbf{true}}$ and $S_{i\mathbf{false}}$ with corresponding $\lambda_{i\mathbf{true}}$ and $\lambda_{i\mathbf{false}}$, such that, for all $\sigma_{\times i}\in S_{i\mathbf{true}}$, $[\![\Psi]\!]_{\sigma_{\times i}}$ is true if $(=_Q)$; and for all $\sigma_{\times i}\in S_{i\mathbf{false}}$, $[\![\Psi]\!]_{\sigma_{\times i}}$ is true if $\neg(=_Q)$.
  Now, for general quantum states ρ_1 and ρ_2 such that $(\langle\sigma_1,\rho_1\rangle,\langle\sigma_2,\rho_2\rangle)\models\Phi\wedge\Theta$, if $\rho_1=\rho_2$, we just follow Argument 2 to construct the coupling from $S_{i\mathbf{true}}$ and $\lambda_{i\mathbf{true}}$ which satisfies $\Psi\wedge\Theta$. If $\rho_1\neq\rho_2$, we also follows similar Argument 2 but construct the coupling from $S_{i\mathbf{false}}$ and $\lambda_{i\mathbf{false}}$ and replace all $(=_Q)$ by $\neg(=_Q)$, so the coupling also satisfies $\Psi\wedge\Theta$.

  Remark A.2. *These argument can also be applied to arbitrary projections rather than global quantum equality; in general, [QFrame] is still sound if we adopt all projections as quantum predicates.*

• [QAdv]. Similar to the proof of [QAdv-Bad] if we set β as $\bot$ and exclude the impossible cases in the proof.

• [QAdv-Bad]. This is a derived rule from other rules.
We first show that, for any $\mathbf{body}_{\mathcal{A}}$,

$$\vdash \mathbf{body}_{\mathcal{A}} \sim \mathbf{body}_{\mathcal{A}} : \beta \Rightarrow \beta. \tag{3}$$

It is a corollary of the fact $\vdash \mathbf{body}_{\mathcal{A}} \sim \mathsf{skip} : \beta \Rightarrow \beta$. We prove it by induction on the structure of $\mathbf{body}_{\mathcal{A}}$ as follows:

- For any basic construct c (skip, deterministic/probabilistic assignment, quantum initialization, unitary transformation and quantum measurement), by premise ast $(\mathbf{body}_{\mathcal{A}})$, so c as a subprogram of $\mathbf{body}_{\mathcal{A}}$ must be lossless, we use [True] and then [PFrame] since free $(\beta) \cap \mathrm{MV}(c) = \emptyset$ (by premises) and obtain $\vdash c \sim c : \top \wedge \beta \Rightarrow \top \wedge \beta$.
- $c \equiv y \leftarrow Q(z, \overline{q})$. Directly by premise $\vdash y \leftarrow Q(z, \overline{q}) \sim \mathsf{skip} : \beta \Rightarrow \beta$.
- $c \equiv c_1; c_2$. Trivial by induction hypotheses and [Seq].
- $c \equiv$ if e then c_1 else c_2. Trivial by induction hypotheses and [Cond-L].
- $c \equiv$ while e do s. By premise ast $(\mathbf{body}_{\mathcal{A}})$, so c as a subprogram of $\mathbf{body}_{\mathcal{A}}$ must be lossless. Then by induction hypotheses and [While-L], we have $\vdash c \sim c : \beta \Rightarrow \beta \wedge \neg e\langle 1\rangle$.

Next, back to [QAdv-Bad], by [QCall], it is sufficient to prove

$$\vdash \mathbf{body}_{\mathcal{A}} \sim \mathbf{body}_{\mathcal{A}} : \Theta \wedge \mathbf{args}_{\mathcal{A}}\langle 1\rangle = \mathbf{args}_{\mathcal{A}}\langle 2\rangle \Rightarrow (\Theta \wedge \mathbf{res}_{\mathcal{A}}\langle 1\rangle = \mathbf{res}_{\mathcal{A}}\langle 2\rangle) \vee \beta.$$

Note that $\Theta \triangleq \mathsf{eqmem}_{\mathcal{A}} \wedge \Psi$ and $\mathsf{eqmem}_{\mathcal{A}}$ asserts the equivalence of all classical variables, so we have:

$$\Theta \wedge \mathbf{args}_{\mathcal{A}}\langle 1\rangle = \mathbf{args}_{\mathcal{A}}\langle 2\rangle \implies \Theta,$$
$$\Theta \vee \beta \implies (\Theta \wedge \mathbf{res}_{\mathcal{A}}\langle 1\rangle = \mathbf{res}_{\mathcal{A}}\langle 2\rangle) \vee \beta.$$

Thus, by [Conseq], it is sufficiently to show

$$\vdash \mathbf{body}_{\mathcal{A}} \sim \mathbf{body}_{\mathcal{A}} : \Theta \Rightarrow \Theta \vee \beta.$$

We prove this by the induction of the structure of $\mathbf{body}_{\mathcal{A}}$.

- For any basic construct c (skip, deterministic/probabilistic assignment, quantum initialization, unitary transformation and quantum measurement), by [Refl] and [PFrame], we have

  $$\vdash c \sim c : \mathsf{eqmem}_{\mathcal{A}} \Rightarrow \mathsf{eqmem}_{\mathcal{A}}.$$

  Since free $(\Psi) \cap \mathrm{MV}(c) = \emptyset$ (by premises), so applying [PFrame] since $\mathrm{cls}(\Psi)$ we have:

  $$\vdash c \sim c : \mathsf{eqmem}_{\mathcal{A}} \wedge \Psi \Rightarrow \mathsf{eqmem}_{\mathcal{A}} \wedge \Psi.$$

  By using [Conseq] and $\Theta \implies \Theta \vee \beta$, we have:

  $$\vdash c \sim c : \Theta \Rightarrow \Theta \vee \beta.$$

- $c \equiv y \leftarrow Q(z, \overline{q})$. We first destruct $\mathsf{eqmem}_{\mathcal{A}} = (=_Q) \wedge \mathsf{eqcmem}\ (S\backslash y) \wedge y\langle 1\rangle = y\langle 2\rangle$. First, by premise, we have

  $$\vdash c \sim c : (=_Q) \wedge z\langle 1\rangle = z\langle 2\rangle \wedge \Psi \wedge \neg\beta \Rightarrow ((=_Q) \wedge y\langle 1\rangle = y\langle 2\rangle \wedge \Psi) \vee \beta.$$

  Since $\mathsf{eqcmem}\ (S\backslash y)$ is a classical assertion and $S\backslash y \cap \mathrm{MV}(y \leftarrow Q(z, \overline{q})) = \emptyset$, we use [PFrame] to obtain

  $$\vdash c \sim c : ((=_Q) \wedge z\langle 1\rangle = z\langle 2\rangle \wedge \Psi \wedge \neg\beta) \wedge \mathsf{eqcmem}\ (S\backslash y) \Rightarrow (((=_Q) \wedge y\langle 1\rangle = y\langle 2\rangle \wedge \Psi) \vee \beta) \wedge \mathsf{eqcmem}\ (S\backslash y).$$

  Realize that

  $$\Theta \wedge \neg\beta \implies ((=_Q) \wedge z\langle 1\rangle = z\langle 2\rangle \wedge \Psi \wedge \neg\beta) \wedge \mathsf{eqcmem}\ (S\backslash y),$$
  $$(((=_Q) \wedge y\langle 1\rangle = y\langle 2\rangle \wedge \Psi) \vee \beta) \wedge \mathsf{eqcmem}\ (S\backslash y) \implies \Theta \vee \beta.$$

  By [Conseq], we have

  $$\vdash c \sim c : \Theta \wedge \neg\beta \Rightarrow \Theta \vee \beta.$$

  According to Eqn. (3) and $\Theta \wedge \beta \implies \beta$, we have

  $$\vdash c \sim c : \Theta \wedge \beta \Rightarrow \beta.$$

  Using [Disj], we get:

  $$\vdash c \sim c : (\Theta \wedge \neg\beta) \vee (\Theta \wedge \beta) \Rightarrow \Theta \vee \beta,$$

  and notice $(\Theta \wedge \neg\beta) \vee (\Theta \wedge \beta) \Leftrightarrow \Theta$.
- $c \equiv c_1; c_2$. By induction hypotheses, we have:

  $$\vdash c_i \sim c_i : \Theta \Rightarrow \Theta \vee \beta \quad \text{for } i = 1, 2.$$

  Besides, according to Eqn. (3) and using [Disj], we have:

  $$\vdash c_2 \sim c_2 : \Theta \vee \beta \Rightarrow (\Theta \vee \beta) \vee \beta.$$

  Then, by [Seq], we obtain $\vdash c \sim c : \Theta \Rightarrow \Theta \vee \beta$.

- $c \equiv \mathsf{if}\ e\ \mathsf{then}\ c_1\ \mathsf{else}\ c_2$. By induction hypotheses, we have:

$$\vdash c_i \sim c_i : \Theta \Rightarrow \Theta \vee \beta \quad \text{for } i = 1, 2.$$

Note that $\Theta \implies e\langle 1\rangle = e\langle 2\rangle$ and $\Theta \wedge e\langle 1\rangle \implies \Theta$ and $\Theta \wedge \neg e\langle 1\rangle \implies \Theta$, by using [Cond], we obtain:

$$\vdash \mathsf{if}\ e\ \mathsf{then}\ c_1\ \mathsf{else}\ c_2 \sim \mathsf{if}\ e\ \mathsf{then}\ c_1\ \mathsf{else}\ c_2 : \Theta \Rightarrow \Theta \vee \beta.$$

- $c \equiv \mathsf{while}\ e\ \mathsf{do}\ s$. By induction hypotheses, we have

$$\vdash s \sim s : \Theta \Rightarrow \Theta \vee \beta.$$

To prove $\vdash c \sim c : \Theta \Rightarrow \Theta \vee \beta$, according to [While-Split], [Struct] and [Conseq], it is sufficient to show:

$$\vdash \mathsf{while}\ e \wedge \neg\beta\ \mathsf{do}\ s; c \sim \mathsf{while}\ e \wedge \neg\beta\ \mathsf{do}\ s; c : \Theta \vee \beta \Rightarrow \Theta \vee \beta.$$

First by the case of β, we notice that

$$\Theta \vee \beta \implies (e\langle 1\rangle \wedge \neg\beta) = (e\langle 2\rangle \wedge \neg\beta).$$

Then apply [While] since $(\Theta \vee \beta) \wedge (e\langle 1\rangle \wedge \neg\beta) \implies \Theta$ and get

$$\vdash \mathsf{while}\ e \wedge \neg\beta\ \mathsf{do}\ s \sim \mathsf{while}\ e \wedge \neg\beta\ \mathsf{do}\ s : \Theta \vee \beta \Rightarrow (\Theta \vee \beta) \wedge \neg(e\langle 1\rangle \wedge \neg\beta),$$

and note that the post-condition is equivalent to $(\Theta \wedge \neg e\langle 1\rangle) \vee \beta$. Next, observe $\Theta \wedge \neg e\langle 1\rangle \implies e\langle 1\rangle = e\langle 2\rangle$ and $(\Theta \wedge \neg e\langle 1\rangle) \wedge e\langle 1\rangle \implies \bot$, and by [False], we have $\vdash s \sim s : \bot \Rightarrow \Theta \wedge \neg e\langle 1\rangle$, then apply [While] again, we get:

$$\vdash c \sim c : \Theta \wedge \neg e\langle 1\rangle \Rightarrow (\Theta \wedge \neg e\langle 1\rangle) \wedge \neg e\langle 1\rangle.$$

According to Eqn. (3) and using [Disj], we have:

$$\vdash c \sim c : (\Theta \wedge \neg e\langle 1\rangle) \vee \beta \Rightarrow ((\Theta \wedge \neg e\langle 1\rangle) \wedge \neg e\langle 1\rangle) \vee \beta.$$

Finally, by [Seq], we obtain

$$\vdash \mathsf{while}\ e \wedge \neg\beta\ \mathsf{do}\ s; c \sim \mathsf{while}\ e \wedge \neg\beta\ \mathsf{do}\ s; c : \Theta \vee \beta \Rightarrow ((\Theta \wedge \neg e\langle 1\rangle) \wedge \neg e\langle 1\rangle) \vee \beta,$$

and complete the proof by noting that post-condition implies $\Theta \vee \beta$.

B A STRENGTHENING OF THE GENERAL LEMMA ON SEMI-CONSTANT DISTRIBUTIONS

B.1 Preliminaries

Notation. Let $\mathcal{X}$ and $\mathcal{Y}$ be two sets such that $\mathcal{X}$ is finite. Given a distribution D over $\mathcal{Y}$, we use $D^{\mathcal{X}}$ do denote the distribution over $\mathcal{X} \mapsto \mathcal{Y}$, where the values associated to each $x \in \mathcal{X}$ are sample independently following the distribution D. We use $x \twoheadleftarrow D$ for sampling a value x according to distribution D. We denote by $\mathbb{B}_\lambda$ the Bernoulli distribution over a single bit $\{0, 1\}$; sampling a bit from $\mathbb{B}_\lambda$ returns 1 with fixed probability λ. Observe that sampling a function f from $\mathbb{B}_\lambda^{\mathcal{X}}$ fixes a set $X_f := \{\, x \in X \,:\, O(x) = 1 \,\} \subseteq \mathcal{X}$. We will overload notation and denote this by $X \twoheadleftarrow \mathbb{B}_\lambda^{\mathcal{X}}$. When A is a quantum algorithm with access to an oracle H, we write $r \twoheadleftarrow A^H$ to denote the measurement of classical output r after a quantum interaction with H, possibly involving many queries.

Theorem B.1 (Markov brothers inequality). *Let P be a polynomial of degree at most n; then, for all nonnegative integers k*

$$\max_{-1 \le x \le 1} \left|P^{(k)}(x)\right| \le \frac{\prod_{i=0}^{k-1}(n^2 - i^2)}{\prod_{i=0}^{k-1} 2i + 1} \max_{-1 \le x \le 1} |P(x)| \ .$$

This implies an immediate corollary.

Corollary B.2. *Let P be a polynomial of degree at most n, such that $P(0) = 0$, $P^{(1)}(0) = 0$ and $\max_{-1 \le x \le 1} |P(x)| \le 1$. For all $x \in \mathbb{R}$, such that $-1 \le x \le 1$, we have*

$$|P(x)| \le \frac{n^4}{6} x^2 \ .$$

Proof. By Theorem B.1, we have $\left|P^{(2)}(x)\right| \le \frac{n^2(n^2-1)}{3} \le \frac{n^4}{3}$. By integrating we have:

$$\left|P^{(1)}(x)\right| = \left|P^{(1)}(x) - P^{(1)}(0)\right| = \left|\int_0^x P^{(2)}(t)\, dt\right| \le \int_0^x \left|P^{(2)}(t)\right| dt \le \int_0^x \frac{n^4}{3}\, dt = \frac{n^4}{3} x$$

By integrating once more, the corrollary follows:

$$|P(x)| = |P(x) - P(0)| = \left|\int_0^x P^{(1)}(t)\, dt\right| \le \int_0^x \left|P^{(1)}(t)\right| dt \le \int_0^x \frac{n^4}{3} t, dt \le \frac{n^4}{6} x^2$$

□

Theorem B.3 (Theorem 3.1 in [54]). *Let A be a quantum algorithm making q quantum queries to an oracle $H : \mathcal{X} \mapsto \mathcal{Y}$ and z a constant bit string. There exists a function $C : \mathcal{X}^{2q} \times \mathcal{Y}^{2q} \times \{0,1\}^* \mapsto \mathbb{R}$ such that, for all distributions D:*

$$\Pr[\, r = z \,:\, H \twoheadleftarrow D^{\mathcal{X}}; r \leftarrow A^{H} \,] = \sum_{\substack{\vec{x} \in \mathcal{X}^{2q} \\ \vec{y} \in \mathcal{Y}^{2q}}} C(\vec{x}, \vec{y}, z) \cdot \Pr[\, \forall i, H(x_i) = y_i \,:\, H \twoheadleftarrow D \,]$$

Definition B.4 (2q-compatibility). Let $F(X)$, for $X \subseteq \mathcal{X}$ be a distribution over oracles of type $\mathcal{X} \mapsto \mathcal{Y}$. We say F is $2q$-compatible if, for all $\vec{x} \in \mathcal{X}^{2q}$, $\vec{y} \in \mathcal{Y}^{2q}$, $X_1, X_2 \subseteq \mathcal{X}$ such that $\forall i, x_i \in X_1 \Leftrightarrow x_i \in X_2$, we have

$$\Pr[\, \forall i, H(x_i) = y_i \,:\, H \twoheadleftarrow F(X_1) \,] = \Pr[\, \forall i, H(x_i) = y_i \,:\, H \twoheadleftarrow F(X_2) \,]$$

B.2 Semi-Constant Distributions

Definition B.5 (Semi-Constant Distribution). Fix a function $H : \mathcal{X} \mapsto \mathcal{Y}$, a set $X \subseteq \mathcal{X}$, and a constant $y \in \mathcal{Y}$. We denote by $\mathrm{SC}_{X,y,H}(x)$ the function returning y if $x \in X$ and $H(x)$ otherwize.

For any λ and distribution D, the semi-constant distribution over $\mathcal{X} \leftarrow \mathcal{Y}$ samples $X \twoheadleftarrow \mathbb{B}_\lambda^{\mathcal{X}}$, $y \twoheadleftarrow D$, and $H \twoheadleftarrow D^{\mathcal{X}}$ and returns $\mathrm{SC}(X, y, H)$. We will often abbreviate this to SC_X, to highlight the conditioning on a pre-sampled set X.

Fix λ and distribution D over $\mathcal{Y}$. We will consider two games G_i, for $i \in \{0,1\}$, where we restrict our attention to quantum algorithms A placing at most q queries to their oracle and that output a bit c, together with some additional information $x \in \mathcal{X}$, $l \in \mathcal{X}^*$. The games are defined as

$$G_i := X \twoheadleftarrow \mathbb{B}_\lambda^{\mathcal{X}}; H \twoheadleftarrow F_i(X); (c, x, l) \twoheadleftarrow A^{H}$$

where $F_0(X) := D^{\mathcal{X}}$, which ignores X, and $F_1(X) := \mathrm{SC}_X$.

Lemma B.6. *F_0 and F_1 are 2q-compatible.*

Proof. For F_0, observe that all inputs are assigned an output according to D, independently of X, so the property holds trivially. For F_1, observe that the distribution of oracle outputs on all inputs is fully determined by whether the input is a member of X or not, and this condition is identical for all elements of $\vec{x}$ in both X_1 and X_2. □

We are interested in *good* executions, which we capture via the following predicate parameterized by an integer k

$$\mathrm{good}_k(X, x, l) := |l| \leq k \wedge x \in X \wedge l \cap X = \emptyset$$

and we define $P_i := \Pr[\, c \wedge \mathrm{good}_k(X, x, l) \,:\, G_i \,]$.

Before stating and proving our main theorem in this section, we state and prove the following auxilliary lemma, where we rely on the fact that both F_0 and F_1 are $2q$-compatible.

Lemma B.7. *For all x^*, l^*, $\vec{x} \in \mathcal{X}^{2q}$ and $\vec{y} \in \mathcal{Y}^{2q}$ such that $|l^*| \leq k$ and $x \notin l^*$, we have that*

$$\Pr[\, \mathrm{good}_k(X, x^*, l^*) \wedge \forall i, H(x_i) = y_i \,:\, X \twoheadleftarrow \mathbb{B}_\lambda^{\mathcal{X}}; H \twoheadleftarrow F_i(X) \,] =$$
$$\textstyle\sum_{X \in \mathcal{P}(O)} \lambda^{|X|+1} (1-\lambda)^{|l^*| + |O \setminus X|} \Pr[\, \forall i, H(x_i) = y_i \,:\, H \twoheadleftarrow F_i(X \cup \{x^*\}) \,]$$

where $O = \{x_i\} \setminus (\{x^*\} \cup l^*)$.

Proof. We first observe that, for both F_0 and F_1, the event we are analysing does not depend on the entire set X, as it only checks X with respect to the fixed values x^*, l^* and $\vec{x}$. This means we can redefine the experiment by sampling X from a distribution $\mathbb{B}_\lambda^{O^*}$ where $O^* = \{x^*, l^*, \vec{x}\}$.

$$\Pr[\, \mathrm{good}_k(X, x^*, l^*) \wedge \forall i, H(x_i) = y_i \,:\, X \twoheadleftarrow \mathbb{B}_\lambda^{\mathcal{X}}; H \twoheadleftarrow F_i(X) \,] =$$
$$\Pr[\, \mathrm{good}_k(X, x^*, l^*) \wedge \forall i, H(x_i) = y_i \,:\, X \twoheadleftarrow \mathbb{B}_\lambda^{O^*}; H \twoheadleftarrow F_i(X) \,]$$

Furthermore, the event only occurs for good X, so we can rewrite the right-hand side of the equality using O as in the theorem statement as follows:

$$\Pr[\, \mathrm{good}(X, x^*, l^*) \,:\, X \twoheadleftarrow \mathbb{B}_\lambda^{O^*\}} \,] \cdot \Pr[\, \forall i, H(x_i) = y_i \,:\, X \twoheadleftarrow \mathbb{B}_\lambda^{O^*}; H \twoheadleftarrow F_i(X) \mid \mathrm{good}(X, x^*, l^*) \,] =$$
$$\lambda \cdot (1-\lambda)^{|l^*|} \cdot \Pr[\, \forall i, H(x_i) = y_i \,:\, X \twoheadleftarrow \mathbb{B}_\lambda^{O}; H \twoheadleftarrow F_i(X \cup \{x^*\}) \,]$$

The lemma follows from rewriting the remaining probability term as the expectation over all X in the support of $\mathbb{B}_\lambda^{O}$ and observing that, for all such X, the probability of X occurring is $\lambda^{\{|X|\}} \cdot (1-\lambda)^{|O \setminus X|}$. □

We now give the main theorem on semi-constant distributions.

Theorem B.8. *Let A be a quantum algorithm making q quantum queries to an oracle $H : \mathcal{X} \mapsto \mathcal{Y}$ returning (c, x, l) where c is a boolean, $x \in \mathcal{X}$ and l is a list of elements in $\mathcal{X}$. We have:*

$$|P_1 - P_0| \leq \frac{(2q + k + 1)^4}{6}\lambda^2$$

Proof. This is a direct consequence of corollary B.2 if we are able to prove that $P = P_1 - P_0$ is a polynomial (in λ) of degree at most $2q + k + 1$ and $P(0) = 0$ and $P^{(1)}(0) = 0$, which we prove next.

Note that, if G_i^X denotes the execution of G conditioned on some set X and $\Pr[X]$ denotes $\Pr[\, X' = X \,:\, X' \twoheadleftarrow \mathbb{B}_\lambda^X \,]$, we can rewrite

$$P_i = \sum_{X \in \mathcal{P}(\mathcal{X})} \Pr[X] \cdot \left(\sum_{\substack{x^* \in \mathcal{X} \\ l^* \in \mathcal{X}^k}} \Pr[\, c \wedge \mathrm{good}_k(X, x, l) \wedge x = x^* \wedge l = l^* \,:\, G_i^X \,] \right).$$

Now it is clear that we can restrict the quantification in the internal summations only to good executions:

$$P_i = \sum_{X \in \mathcal{P}(\mathcal{X})} \Pr[X] \cdot \left(\sum_{\substack{x^* \in X \\ l^* \in (\mathcal{X} \setminus X)^k}} \Pr[\, c \wedge x = x^* \wedge l = l^* \,:\, G_i^X \,] \right).$$

Now each probability term in the internal summation can be matched to to the left-hand side of Theorem B.3, as we are checking the output of a quantum algorithm, which is interacting with an oracle sampled from some distribution, with respect to a constant value. The summation can therefore be rewritten as.

$$P_i = \sum_{X \in \mathcal{P}(\mathcal{X})} \Pr[X] \cdot \left(\sum_{\substack{x^* \in X \\ l^* \in (\mathcal{X} \setminus X)^k}} \sum_{\substack{\vec{x} \in \mathcal{X}^{2q} \\ \vec{y} \in \mathcal{Y}^{2q}}} C(\vec{x}, \vec{y}, (1, x^*, l^*)) \cdot \Pr[\, \forall i, H(x_i) = y_i \,:\, H \twoheadleftarrow F_i(X) \,] \right).$$

We can now bring the summation over input/output pairs outside.

$$P_i = \sum_{\substack{\vec{x} \in \mathcal{X}^{2q} \\ \vec{y} \in \mathcal{Y}^{2q}}} \sum_{X \in \mathcal{P}(\mathcal{X})} \Pr[X] \cdot \left(\sum_{\substack{x^* \in X \\ l^* \in (\mathcal{X} \setminus X)^k}} C(\vec{x}, \vec{y}, (1, x^*, l^*)) \cdot \Pr[\, \forall i, H(x_i) = y_i \,:\, H \twoheadleftarrow F_i(X) \,] \right).$$

To bring the summation over X inside, we must rewrite the internal summations as follows:

$$P_i = \sum_{\substack{\vec{x} \in \mathcal{X}^{2q} \\ \vec{y} \in \mathcal{Y}^{2q} \\ x^* \in \mathcal{X} \\ l^* \in (\mathcal{X} \setminus \{x\})^k}} C(\vec{x}, \vec{y}, (1, x^*, l^*)) \cdot \left(\sum_{X \in \mathcal{P}(\mathcal{X})} \Pr[X \wedge \mathrm{good}_k(X, x^*, l^*)] \cdot \Pr[\, \forall i, H(x_i) = y_i \,:\, H \twoheadleftarrow F_i(X) \,] \right).$$

Which in turn yields

$$P_i = \sum_{\substack{\vec{x} \in \mathcal{X}^{2q} \\ \vec{y} \in \mathcal{Y}^{2q} \\ x^* \in \mathcal{X} \\ l^* \in (\mathcal{X} \setminus \{x\})^k}} C(\vec{x}, \vec{y}, (1, x^*, l^*)) \cdot \left(\sum_{X \in \mathcal{P}(\mathcal{X})} \Pr[\, X' = X \wedge \mathrm{good}_k(X, x^*, l^*) \wedge \forall i, H(x_i) = y_i \,:\, X' \twoheadleftarrow \mathbb{B}_\lambda^{\mathcal{X}}; H \twoheadleftarrow F_i(X) \,] \right).$$

Which in turn yields

$$P_i = \sum_{\substack{\vec{x} \in \mathcal{X}^{2q} \\ \vec{y} \in \mathcal{Y}^{2q} \\ x^* \in \mathcal{X} \\ l^* \in (\mathcal{X} \setminus \{x\})^k}} C(\vec{x}, \vec{y}, (1, x^*, l^*)) \cdot \Pr[\, \mathrm{good}_k(X, x^*, l^*) \wedge \forall i, H(x_i) = y_i \,:\, X \twoheadleftarrow \mathbb{B}_\lambda^{\mathcal{X}}; H \twoheadleftarrow F_i(X) \,] \,.$$

Now let $R^{F_i}(x^*, l^*, \vec{x}, \vec{y})$ be the polynomial

$$\sum_{X \in \mathcal{P}(O), X \neq \emptyset} \lambda^{|X|-1}(1-\lambda)^{|l^*|+|O \setminus X|} \Pr[\, \forall i, H(x_i) = y_i \,:\, H \twoheadleftarrow F_i(X \cup \{x^*\}) \,]$$

which is of degree at most $2q + k - 1$. Applying Lemma B.7, have that

$$\Pr[\, \mathrm{good}_k(X, x^*, l^*) \wedge \forall i, H(x_i) = y_i \,:\, X \twoheadleftarrow \mathbb{B}_\lambda^{\mathcal{X}}; H \twoheadleftarrow F_i(X) \,] = \Pr[\, \forall i, H(x_i) = y_i \,:\, H \twoheadleftarrow F_i(\{x^*\}) \,] + \lambda^2 R^{F_i}(x^*, l^*, \vec{x}, \vec{y})$$

Putting it all together we obtain

$$P_i = \sum_{\substack{\vec{x} \in \mathcal{X}^{2q} \\ \vec{y} \in \mathcal{Y}^{2q} \\ x^* \in \mathcal{X} \\ l^* \in (\mathcal{X} \setminus \{x\})^k}} C(\vec{x}, \vec{y}, (1, x^*, l^*)) \cdot \left(\Pr[\, \forall i, H(x_i) = y_i \,:\, H \twoheadleftarrow F_i(\{x^*\}) \,] + \lambda^2 R^{F_i}(x^*, l^*, \vec{x}, \vec{y}) \right) \,.$$

We now note that the distributions $F_1(\{x^*\})$ and $F_0(\{x^*\})$ are identical. So

$$\Pr[\, \forall i, H(x_i) = y_i \,:\, H \twoheadleftarrow F_0(\{x^*\}) \,] = \Pr[\, \forall i, H(x_i) = y_i \,:\, H \twoheadleftarrow F_1(\{x^*\}) \,]$$

And finally

$$P_1 - P_0 = \sum_{\substack{\vec{x} \in \mathcal{X}^{2q} \\ \vec{y} \in \mathcal{Y}^{2q} \\ x^* \in \mathcal{X} \\ l^* \in (\mathcal{X} \setminus \{x\})^k}} C(\vec{x}, \vec{y}, (1, x^*, l^*)) \cdot \lambda^2 \cdot \left(R^{F_1}(x^*, l^*, \vec{x}, \vec{y}) - R^{F_0}(x^*, l^*, \vec{x}, \vec{y}) \right) \,.$$

Since $P = P_1 - P_0$ is a polynomial in λ of degree at most $2q + k + 1$ and $P(0) = 0$ and $P^{(1)}(0) = 0$, we have by Corollary B.2 that

$$|P_1 - P_0| \leq \frac{(2q + k + 1)^4}{6} \lambda^2$$

□