A Core Calculus for Equational Proofs of Distributed Cryptographic Protocols: Technical Report

February 15, 2023

Acknowledgement

This project was funded through the NGI Assure Fund, a fund established by NLnet with financial support from the European Commission's Next Generation Internet programme, under the aegis of DG Communications Networks, Content and Technology under grant agreement No. 957073.

1 Syntax of IPDL

IPDL is built from three layers: *protocols* are networks of mutually interacting *reactions*, each of which is a simple monadic, probabilistic program that computes an *expression*. In the context of a protocol, a reaction operates on a unique *channel* and may read from other channels, thereby utilizing expressions computed by other reactions. The syntax and judgements of IPDL are outlined in Figures 1, 2, respectively, and are parameterized by a user-defined *signature* Σ:

Definition 1 (Signature)**.** *An IPDL signature Σ is a finite collection of:*

- *type symbols* t*;*
- *typed function symbols* $\mathsf{f} : \tau \to \sigma$*; and*
- *typed distribution symbols* $\mathsf{d} : \tau \twoheadrightarrow \sigma$*.*

We have a minimal set of data types, including the unit type $\mathsf{1}$, Booleans, products, as well as arbitrary type symbols t, drawn from the signature Σ. Expressions are used for non-probabilistic computations, and are standard. All values in IPDL are bitstrings of a length given by data types, so we annotate the operations $\mathsf{fst}_{\tau\times\sigma}$ and $\mathsf{snd}_{\tau\times\sigma}$ with the type of the pair to determine the index to split the pair into two; for readability we omit this subscript whenever appropriate. Function symbols f must be declared in the signature Σ, and for a constant $\mathsf{f} : \mathsf{1} \to \tau$, we write f in place of $\mathsf{f}\ \checkmark$. Substitutions $\theta : \Gamma_1 \to \Gamma_2$ between type contexts are standard.

As mentioned above, reactions are monadic programs which may return expressions, sample from distributions, read from channels, branch on a value of type Bool, and sequentially compose. Analogously to function symbols, distribution symbols d must be declared in the signature Σ, and for a constant $\mathsf{d} : \mathsf{1} \twoheadrightarrow \tau$, we write $\mathsf{samp}\ \mathsf{d}$ instead of $\mathsf{samp}\ (\mathsf{d}\ \checkmark)$. For readability, we often omit the type of the bound variable in a sequential composition, and write $x \leftarrow \mathsf{read}\ c;\ R$ and $x \leftarrow \mathsf{samp}\ \mathsf{d};\ R$ simply as $x \leftarrow c;\ R$ and $x \leftarrow \mathsf{d};\ R$ wherever appropriate. Protocols in IPDL are given by a simple but expressive syntax: channel assignment $o := R$ assigns the reaction R to channel o; parallel composition $P \mid\mid Q$ allows P and Q to freely interact concurrently; and channel generation $\mathsf{new}\ o : \tau\ \mathsf{in}\ P$ creates a new, internal channel for use in P. *Embeddings* $\phi : \Delta_1 \to \Delta_2$ between channel contexts are injective, type-preserving mappings specifying how to rename channels in Δ_2 to fit in the larger context Δ_1.

1.1 Typing

We restrict our attention to well-typed IPDL constructs. In addition to respecting data types, the typing judgments guarantee that all reads from channels in reactions are in scope, and that all channels are assigned at most one reaction in protocols. The typing $\Gamma \vdash e : \tau$ for expressions is standard, see Figure 3. Figure 4 shows the typing

$$
\begin{array}{lll}
\text{Data Types} & \tau, \sigma & ::= \ \mathsf{t} \mid \mathsf{1} \mid \mathsf{Bool} \mid \tau \times \tau \\
\text{Expressions} & e & ::= \ x \mid \checkmark \mid \mathsf{true} \mid \mathsf{false} \mid \mathsf{f}\ e \mid (e_1, e_2) \mid \mathsf{fst}_{\tau\times\sigma}\ e \mid \mathsf{snd}_{\tau\times\sigma}\ e \\
\text{Channels} & i, o, c & \\
\text{Reactions} & R, S & ::= \ \mathsf{ret}\ e \mid \mathsf{samp}\ (\mathsf{d}\ e) \mid \mathsf{read}\ c \mid \mathsf{if}\ e\ \mathsf{then}\ R_1\ \mathsf{else}\ R_2 \mid x : \sigma \leftarrow R;\ S \\
\text{Protocols} & P, Q & ::= \ 0 \mid o \coloneqq R \mid P \mid\mid Q \mid \mathsf{new}\ o : \tau\ \mathsf{in}\ P \\
\text{Channel Sets} & I, O & ::= \ \{c_1, \ldots, c_n\} \\
\text{Type Contexts} & \Gamma & ::= \ \cdot \mid \Gamma, x : \tau \\
\text{Channel Contexts} & \Delta & ::= \ \cdot \mid \Delta, c : \tau
\end{array}
$$

Figure 1: Syntax of IPDL .

$$
\begin{array}{ll}
\text{Expression Typing} & \Gamma \vdash e : \tau \\
\text{Reaction Typing} & \Delta;\ \Gamma \vdash R : I \to \tau \\
\text{Protocol Typing} & \Delta \vdash P : I \to O \\
 & \\
\text{Substitutions} & \theta : \Gamma_1 \to \Gamma_2 \\
\text{Embeddings} & \phi : \Delta_1 \to \Delta_2 \\
 & \\
\text{Expression Equality} & \Gamma \vdash e_1 = e_2 : \tau \\
\text{Reaction Equality} & \Delta;\ \Gamma \vdash R_1 = R_2 : I \to \tau \\
\text{Protocol Equality (Strict)} & \Delta \vdash P_1 = P_2 : I \to O
\end{array}
$$

Figure 2: Judgements of the exact fragment of ipdl.

rules for reactions. Intuitively, $\Delta;\ \Gamma \vdash R : I \to \tau$ holds when R uses variables in Γ, reads from channels in I typed according to Δ, and returns a value of type τ. Figure 5 gives the typing rules for protocols: $\Delta \vdash P : I \to O$ holds when P uses inputs in I to assign reactions to the channels in O, all typed according to Δ.

Channel assignment $o \coloneqq R$ has the type $I \to \{o\}$ when R is well-typed with an empty variable context, making use of inputs from I as well as of o. We allow R to read from its own output o to express divergence: the protocol $o \coloneqq \mathsf{read}\ o$ cannot reduce, which is useful for (conditionally) deactivating certain outputs. The typing rule for parallel composition $P \mid\mid Q$ states that P may use the outputs of Q as inputs while defining its own outputs, and vice versa. Importantly, the typing rules ensure that the outputs of P and Q are disjoint so that each channel carries a unique reaction. Finally, the rule for channel generation allows a protocol to select a fresh channel name o, assign it a type τ, and use it for internal computation and communication. Protocol typing plays a crucial role for modeling security. Simulation-based security in IPDL is modeled by the existence of a *simulator* Sim with an appropriate typing judgment, $\Delta \vdash \mathsf{Sim} : I \to O$. Restricting the behavior of Sim to only use inputs along I is necessary to rule out trivial results (*e.g.*, Sim simply copies a secret from the specification).

$$
\boxed{\Gamma \vdash e : \tau}
$$

$$
\frac{x : \tau \in \Gamma}{\Gamma \vdash x : \tau} \qquad \frac{}{\Gamma \vdash \checkmark : 1} \qquad \frac{}{\Gamma \vdash \mathsf{true} : \mathsf{Bool}} \qquad \frac{}{\Gamma \vdash \mathsf{false} : \mathsf{Bool}} \qquad \frac{\mathsf{f} : \sigma \to \tau \in \Sigma \qquad \Gamma \vdash e : \sigma}{\Gamma \vdash \mathsf{f}\ e : \tau}
$$

$$
\frac{\Gamma \vdash e_1 : \tau_1 \qquad \Gamma \vdash e_2 : \tau_2}{\Gamma \vdash (e_1, e_2) : \tau_1 \times \tau_2} \qquad \frac{\Gamma \vdash e : \sigma \times \tau}{\Gamma \vdash \mathsf{fst}_{\sigma\times\tau}\ e : \sigma} \qquad \frac{\Gamma \vdash e : \sigma \times \tau}{\Gamma \vdash \mathsf{snd}_{\sigma\times\tau}\ e : \tau}
$$

Figure 3: Typing for IPDL expressions.

$$\boxed{\Delta;\ \Gamma \vdash R : I \to \tau}$$

$$\frac{\Gamma \vdash e : \tau}{\Delta;\ \Gamma \vdash \mathsf{ret}\ e : I \to \tau} \qquad \frac{\mathsf{d} : \sigma \twoheadrightarrow \tau \in \Sigma \qquad \Gamma \vdash e : \sigma}{\Delta;\ \Gamma \vdash \mathsf{samp}\ (\mathsf{d}\ e) : I \to \tau} \qquad \frac{(i : \tau) \in \Delta \qquad i \in I}{\Delta;\ \Gamma \vdash \mathsf{read}\ i : I \to \tau}$$

$$\frac{\Gamma \vdash e : \mathsf{Bool} \qquad \Delta;\ \Gamma \vdash R_1 : I \to \tau \qquad \Delta;\ \Gamma \vdash R_2 : I \to \tau}{\Delta;\ \Gamma \vdash \mathsf{if}\ e\ \mathsf{then}\ R_1\ \mathsf{else}\ R_2 : I \to \tau} \qquad \frac{\Delta;\ \Gamma \vdash R : I \to \sigma \qquad \Delta;\ \Gamma, x : \sigma \vdash S : I \to \tau}{\Delta;\ \Gamma \vdash (x : \sigma \leftarrow R;\ S) : I \to \tau}$$

Figure 4: Typing for IPDL reactions.

$$\boxed{\Delta \vdash P : I \to O}$$

$$\frac{}{\Delta \vdash 0 : I \to \varnothing} \qquad \frac{o : \tau \in \Delta \qquad o \notin I \qquad \Delta;\ \cdot \vdash R : I \cup \{o\} \to \tau}{\Delta \vdash (o := R) : I \to \{o\}}$$

$$\frac{\Delta \vdash P : I \cup O_2 \to O_1 \qquad \Delta \vdash Q : I \cup O_1 \to O_2}{\Delta \vdash P \parallel Q : I \to O_1 \cup O_2} \qquad \frac{\Delta, o : \tau \vdash P : I \to O \cup \{o\}}{\Delta \vdash (\mathsf{new}\ o : \tau\ \mathsf{in}\ P) : I \to O}$$

Figure 5: Typing for IPDL protocols.

1.2 Equational Logic

We now present the equational logic of IPDL . As mentioned above, the logic is divided into *exact* rules that establish semantic equivalences between protocols, and *approximate* rules that are used to discharge computational indistinguishability assumptions.

1.2.1 Exact Equality

The bulk of the reasoning in IPDL is done using exact equalities. At the expression level, we assume an ambient finite set of axioms of the form $\Gamma \vdash e_1 = e_2 : \tau$, where $\Gamma \vdash e_1 : \tau$ and $\Gamma \vdash e_2 : \tau$. The rules for expression equality are standard, see Figure 6.

At the reaction level, we analogously assume an ambient finite set of axioms of the form $\Delta;\ \Gamma \vdash R_1 = R_2 : I \to \tau$, where $\Delta;\ \Gamma \vdash R_1 : I \to \tau$ and $\Delta;\ \Gamma \vdash R_2 : I \to \tau$. The rules for reaction equality, shown in Figures 7, 8, ensure in particular that reactions form a *commutative monad*: we have

$$\big(x \leftarrow R_1;\ y \leftarrow R_2;\ S(x,y)\big) = \big(y \leftarrow R_2;\ x \leftarrow R_1;\ S(x,y)\big)$$

whenever R_2 does not depend on x. All expected equivalences for commutative monads hold for reactions, including the usual monad laws and congruence of equivalence under monadic bind. The SAMP-PURE rule allows us to drop an unused sampling, and the READ-DET rule allows us to replace two reads from the same channel by a single one. The rules IF-LEFT, IF-RIGHT, and IF-EXT allow us to manipulate conditionals.

At the protocol level, we similarly assume an ambient finite set of axioms of the form $\Delta \vdash P_1 = P_2 : I \to O$, where $\Delta \vdash P_1 : I \to O$ and $\Delta \vdash P_2 : I \to O$. We use these axioms to specify user-defined functional assumptions, *e.g.*, the correctness of decryption. Exact protocol equivalences allow us to reason about communication between subprotocols and functional correctness, and to simplify intermediate computations. We will see later that exact equivalence implies the existence of a *bisimulation* on protocols, which in turn implies perfect computational indistinguishability against an arbitrary distinguisher. The rules for the exact equality of protocols are in Figures 9, 10; we now describe them informally.

The COMP-NEW rule allows us to permute parallel composition and the creation of a new channel, and the same as *scope extrusion* in process calculi [**?**]. The ABSORB-LEFT rule allows us to discard a component in a

parallel composition if it has no outputs; this allows us to eliminate internal channels once they are no longer used. The DIVERGE rule allows us to simplify diverging reactions: if a channel reads from itself and continues as an arbitrary reaction R, then we can safely discard R as we will never reach it in the first place. The three (un)folding rules FOLD-IF-LEFT, FOLD-IF-RIGHT, and FOLD-BIND allow us to simplify composite reactions by bringing their components into the protocol level as separate internal channels. The rule SUBSUME states that channel dependency is transitive: if we depend on o_1, and o_1 in turn depends on o_0, then we also depend on o_0, and this dependency can be made explicit. The SUBST rule allows us to inline certain reactions into read commands. Inlining $o_1 \coloneqq R_1$ into $o_2 \coloneqq x \leftarrow \mathsf{read}\ o_1;\ R_2$ is sound provided R_1 is *duplicable*: observing two independent results of evaluating R_1 is equivalent to observing the same result twice. This side condition is easily discharged whenever R_1 does not contain probabilistic sampling. Finally, the DROP rule allows dropping unused reads from channels in certain situations. Due to timing dependencies among channels, we only allow dropping reads from the channel $o_1 \coloneqq R_1$ in the context of $o_2 \coloneqq \_ \leftarrow \mathsf{read}\ o_1;\ R_2$ when we have that $(\_ \leftarrow R_1;\ R_2) = R_2$. This side condition is met whenever all reads present in R_1 are also present in R_2.

$$\boxed{\Gamma \vdash e_1 = e_2 : \tau}$$

$$\frac{\Gamma \vdash e : \tau}{\Gamma \vdash e = e : \tau}\ \textsc{refl} \qquad \frac{\Gamma \vdash e_1 = e_2 : \tau}{\Gamma \vdash e_2 = e_1 : \tau}\ \textsc{sym} \qquad \frac{\Gamma \vdash e_1 = e_2 : \tau \qquad \Gamma \vdash e_2 = e_3 : \tau}{\Gamma \vdash e_1 = e_3 : \tau}\ \textsc{trans}$$

$$\frac{\Gamma \vdash e_1 = e_2 : \tau\ \mathsf{axiom}}{\Gamma \vdash e_1 = e_2 : \tau}\ \textsc{axiom} \qquad \frac{\theta : \Gamma_1 \to \Gamma_2 \qquad \Gamma_2 \vdash e_1 = e_2 : \tau}{\Gamma_1 \vdash \theta^\star(e_1) = \theta^\star(e_2) : \tau}\ \textsc{subst}$$

$$\frac{\mathsf{f} : \sigma \to \tau \in \Sigma \qquad \Gamma \vdash e = e' : \sigma}{\Gamma \vdash \mathsf{f}\ e = \mathsf{f}\ e' : \tau}\ \textsc{app-cong} \qquad \frac{\Gamma \vdash e_1 = e_1' : \tau_1 \qquad \Gamma \vdash e_2 = e_2' : \tau_2}{\Gamma \vdash (e_1, e_2) = (e_1', e_2') : \tau_1 \times \tau_2}\ \textsc{pair-cong}$$

$$\frac{\Gamma \vdash e = e' : \sigma \times \tau}{\Gamma \vdash \mathsf{fst}_{\sigma\times\tau}\ e = \mathsf{fst}_{\sigma\times\tau}\ e' : \sigma}\ \textsc{fst-cong} \qquad \frac{\Gamma \vdash e = e' : \sigma \times \tau}{\Gamma \vdash \mathsf{snd}_{\sigma\times\tau}\ e = \mathsf{snd}_{\sigma\times\tau}\ e' : \tau}\ \textsc{snd-cong}$$

$$\frac{\Gamma \vdash e_1 : \tau_1 \qquad \Gamma \vdash e_2 : \tau_2}{\Gamma \vdash \mathsf{fst}_{\tau_1\times\tau_2}\ (e_1, e_2) = e_1 : \tau_1}\ \textsc{fst-pair} \qquad \frac{\Gamma \vdash e_1 : \tau_1 \qquad \Gamma \vdash e_2 : \tau_2}{\Gamma \vdash \mathsf{snd}_{\tau_1\times\tau_2}\ (e_1, e_2) = e_2 : \tau_2}\ \textsc{snd-pair}$$

$$\frac{\Gamma \vdash e : \sigma \times \tau}{\Gamma \vdash e = \left(\mathsf{fst}_{\sigma\times\tau}\ e,\ \mathsf{snd}_{\sigma\times\tau}\ e\right) : \sigma \times \tau}\ \textsc{pair-ext} \qquad \frac{\Gamma \vdash e : 1}{\Gamma \vdash e = \checkmark : 1}\ \textsc{one-ext}$$

Figure 6: Equality for IPDL expressions.

1.2.2 Approximate Equality

The equational theory for the approximate fragment of IPDL consists of two layers: one for the *approximate equality* of protocols, and one for the *asymptotic equality* of protocol families as functions of the security parameter $\lambda \in \mathbb{N}$. The approximate equality judgement $\Delta \vdash P \approx Q : I \to O\ \mathsf{width}\ k\ \mathsf{length}\ l$ equates two protocols $\Delta \vdash P : I \to O$ and $\Delta \vdash Q : I \to O$ with identical typing judgements. We think of these as corresponding to a specific security parameter λ. Analogously to exact protocol equality, we assume an ambient finite set of *approximate axioms* of the form $\Delta \vdash P \approx Q : I \to O$, where $\Delta \vdash P : I \to O$ and $\Delta \vdash Q : I \to O$. These axioms capture cryptographic assumptions on computational indistinguishability.

The parameters $k, l \in \mathbb{N}$ track the size of the derivation. The *width* parameter k simply counts the number of invocations of axioms applied during the proof: applying a single approximate axiom incurs $k = 1$, and we sum up the two values of k whenever we use transitivity. In the asymptotic equality judgement, k becomes a function of the security parameter λ and we require that it be bounded by a polynomial in λ: even though each individual axiom invocation introduces a negligible error, the sum of exponentially many negligible errors may not be negligible

$$\boxed{\Delta;\ \Gamma \vdash R_1 = R_2 : I \to \tau}$$

$$\frac{\Delta;\ \Gamma \vdash R : I \to \tau}{\Delta;\ \Gamma \vdash R = R : I \to \tau}\ \textsc{refl} \qquad \frac{\Delta;\ \Gamma \vdash R_1 = R_2 : I \to \tau}{\Delta;\ \Gamma \vdash R_2 = R_1 : I \to \tau}\ \textsc{sym}$$

$$\frac{\Delta;\ \Gamma \vdash R_1 = R_2 : I \to \tau \qquad \Delta;\ \Gamma \vdash R_2 = R_3 : I \to \tau}{\Delta;\ \Gamma \vdash R_1 = R_3 : I \to \tau}\ \textsc{trans} \qquad \frac{\Delta;\ \Gamma \vdash R_1 = R_2 : I \to \tau\ \mathsf{axiom}}{\Delta;\ \Gamma \vdash R_1 = R_2 : I \to \tau}\ \textsc{axiom}$$

$$\frac{\theta : \Gamma_1 \to \Gamma_2 \qquad \Delta;\ \Gamma_2 \vdash R_1 = R_2 : I \to \tau}{\Delta;\ \Gamma_1 \vdash \theta^\star(R_1) = \theta^\star(R_2) : I \to \tau}\ \textsc{subst} \qquad \frac{\phi : \Delta_1 \to \Delta_2 \qquad \Delta_2;\ \Gamma \vdash R_1 = R_2 : I \to \tau}{\Delta_1;\ \Gamma \vdash \phi^\star(R_1) = \phi^\star(R_2) : \phi^\star(I) \to \tau}\ \textsc{embed}$$

$$\frac{i \notin I \qquad \Delta;\ \Gamma \vdash R_1 = R_2 : I \to \tau}{\Delta;\ \Gamma \vdash R_1 = R_2 : I \cup \{i\} \to \tau}\ \textsc{input-unused} \qquad \frac{\Gamma \vdash e = e' : \tau}{\Delta;\ \Gamma \vdash \mathsf{ret}\ e = \mathsf{ret}\ e' : I \to \tau}\ \textsc{cong-ret}$$

$$\frac{\mathsf{d} : \sigma \twoheadrightarrow \tau \in \Sigma \qquad \Gamma \vdash e = e' : \sigma}{\Delta;\ \Gamma \vdash \mathsf{samp}\ (\mathsf{d}\ e) = \mathsf{samp}\ (\mathsf{d}\ e') : I \to \tau}\ \textsc{cong-samp}$$

$$\frac{\Gamma \vdash e = e' : \mathsf{Bool} \qquad \Delta;\ \Gamma \vdash R_1 = R_1' : I \to \tau \qquad \Delta;\ \Gamma \vdash R_2 = R_2' : I \to \tau}{\Delta;\ \Gamma \vdash \big(\mathsf{if}\ e\ \mathsf{then}\ R_1\ \mathsf{else}\ R_2\big) = \big(\mathsf{if}\ e'\ \mathsf{then}\ R_1'\ \mathsf{else}\ R_2'\big) : I \to \tau}\ \textsc{cong-if}$$

$$\frac{\Delta;\ \Gamma \vdash R = R' : I \to \sigma \qquad \Delta;\ \Gamma, x : \sigma \vdash S = S' : I \to \tau}{\Delta;\ \Gamma \vdash (x : \sigma \leftarrow R;\ S) = (x : \sigma \leftarrow R';\ S') : I \to \tau}\ \textsc{cong-bind}$$

Figure 7: Equality for IPDL reactions. Additional rules are given in Figure 8.

anymore.

Since most nontrivial reasoning in IPDL is done in the exact half, the approximate equality rules are used mostly to apply indistinguishability assumptions nested deeply inside protocols. The *length* parameter l tracks the largest size of such a nesting – also known as a *program context*. In the asymptotic equality judgement, we again require that l be bounded as a function of λ by a polynomial: exponentially large IPDL contexts could in principle be used to encode exponential-time probabilistic computations. An IPDL program context surrounding an indistinguishability assumption is formally a part of the adversary, and as such it must be resource-bounded for the indistinguishability assumption to apply.

In IPDL, the bound on resources is given by a *symbolic size* function $|\cdot|$ defined for expressions, reactions, and protocols. Since we assume that all function symbols will be interpreted by functions computable in poly-time, our symbolic size for expressions simply counts the number of variables and function applications present:

$$\begin{aligned}
|x| &:= 1 \\
|\checkmark| &:= 1 \\
|\mathsf{true}| &:= 1 \\
|\mathsf{false}| &:= 1 \\
|\mathsf{f}\ e| &:= |e| + 1 \\
|(e_1, e_2)| &:= |e_1| + |e_2| \\
|\mathsf{fst}_{\sigma\times\tau}\ e| &:= |e| \\
|\mathsf{snd}_{\sigma\times\tau}\ e| &:= |e|
\end{aligned}$$

For reactions, we follow a similar principle: we assume that all distribution symbols will be interpreted by probabilistic functions approximately computable in poly-time, and count the number of variables and (probabilistic)

$$\boxed{\Delta;\ \Gamma \vdash R_1 = R_2 : I \to \tau}$$

$$\frac{\Gamma \vdash e : \sigma \qquad \Delta;\ \Gamma, x : \sigma \vdash R : I \to \tau}{\Delta;\ \Gamma \vdash (x : \sigma \leftarrow \mathsf{ret}\ e;\ R) = R[x := e] : I \to \tau}\ \textsc{ret-bind} \qquad \frac{\Delta;\ \Gamma \vdash R : I \to \tau}{\Delta;\ \Gamma \vdash (x : \tau \leftarrow R;\ \mathsf{ret}\ x) = R : I \to \tau}\ \textsc{bind-ret}$$

$$\frac{\Delta;\ \Gamma \vdash R_1 : I \to \sigma_1 \qquad \Delta;\ \Gamma, x_1 : \sigma_1 \vdash R_2 : I \to \sigma_2 \qquad \Delta;\ \Gamma, x_2 : \sigma_2 \vdash S : I \to \tau}{\Delta;\ \Gamma \vdash \big(x_2 : \sigma_2 \leftarrow (x_1 : \sigma_1 \leftarrow R_1;\ R_2);\ S\big) = \big(x_1 : \sigma_1 \leftarrow R_1;\ x_2 : \sigma_2 \leftarrow R_2;\ S\big) : I \to \tau}\ \textsc{bind-bind}$$

$$\frac{\Delta;\ \Gamma \vdash R_1 : I \to \sigma_1 \qquad \Delta;\ \Gamma \vdash R_2 : I \to \sigma_2 \qquad \Delta;\ \Gamma, x_1 : \sigma_1, x_2 : \sigma_2 \vdash S : I \to \tau}{\Delta;\ \Gamma \vdash \big(x_1 : \sigma_1 \leftarrow R_1;\ x_2 : \sigma_2 \leftarrow R_2;\ S\big) = \big(x_2 : \sigma_2 \leftarrow R_2;\ x_1 : \sigma_1 \leftarrow R_1;\ S\big) : I \to \tau}\ \textsc{exch}$$

$$\frac{\mathsf{d} : \sigma_1 \twoheadrightarrow \sigma_2 \in \Sigma \qquad \Gamma \vdash e : \sigma_1 \qquad \Delta;\ \Gamma \vdash R : I \to \tau}{\Delta;\ \Gamma \vdash \big(x : \sigma_2 \leftarrow \mathsf{samp}\ (\mathsf{d}\ e);\ R\big) = R : I \to \tau}\ \textsc{samp-pure}$$

$$\frac{i : \sigma \in \Delta \qquad i \in I \qquad \Delta;\ \Gamma, x : \sigma, y : \sigma \vdash R : I \to \tau}{\Delta;\ \Gamma \vdash \big(x : \sigma \leftarrow \mathsf{read}\ i;\ y : \sigma \leftarrow \mathsf{read}\ i;\ R\big) = \big(x : \sigma \leftarrow \mathsf{read}\ i;\ R[y := x]\big) : I \to \tau}\ \textsc{read-det}$$

$$\frac{\Delta;\ \Gamma \vdash R_1 : I \to \tau \qquad \Delta;\ \Gamma \vdash R_2 : I \to \tau}{\Delta;\ \Gamma \vdash \big(\mathsf{if}\ \mathsf{true}\ \mathsf{then}\ R_1\ \mathsf{else}\ R_2\big) = R_1 : I \to \tau}\ \textsc{if-left}$$

$$\frac{\Delta;\ \Gamma \vdash R_1 : I \to \tau \qquad \Delta;\ \Gamma \vdash R_2 : I \to \tau}{\Delta;\ \Gamma \vdash \big(\mathsf{if}\ \mathsf{false}\ \mathsf{then}\ R_1\ \mathsf{else}\ R_2\big) = R_2 : I \to \tau}\ \textsc{if-right}$$

$$\frac{\Delta;\ \Gamma, x : \mathsf{Bool} \vdash R : I \to \tau \qquad \Gamma \vdash e : \mathsf{Bool}}{\Delta;\ \Gamma \vdash \big(\mathsf{if}\ e\ \mathsf{then}\ R[x := \mathsf{true}]\ \mathsf{else}\ R[x := \mathsf{false}]\big) = R[x := e] : I \to \tau}\ \textsc{if-ext}$$

Figure 8: Equality for IPDL reactions.

function applications:

$$\begin{aligned} |\mathsf{ret}\ e| &:= |e| \\ |\mathsf{samp}\ (\mathsf{d}\ e)| &:= |e| + 1 \\ |\mathsf{read}\ c| &:= 1 \\ |\mathsf{if}\ e\ \mathsf{then}\ R_1\ \mathsf{else}\ R_2| &:= |e| + \mathsf{max}\,(|R_1|, |R_2|) \\ |x : \sigma \leftarrow R;\ S| &:= |R| + |S| \end{aligned}$$

The only slightly subtle rule here is for branching, where we take the greater of the two bounds for the two branches, and add it to the bound for the condition. Since protocols in IPDL are finite networks of channels that do not contain recursion, the bound for a protocol is simply the sum of the bounds of all reactions occurring in the protocol:

$$\begin{aligned} |0| &:= 0 \\ |o := R| &:= |R| \\ |P \mid\mid Q| &:= |P| + |Q| \\ |\mathsf{new}\ o : \tau\ \mathsf{in}\ P| &:= |P| \end{aligned}$$

Figure 11 shows the rules for the approximate equality of IPDL protocols; crucially, rule STRICT allows us to descend to the exact half of the proof system. Whenever we need to make the ambient theory with approximate axioms $\Delta^1 \vdash P^1 \approx Q^1 : I^1 \to O^1, \ldots, \Delta^n \vdash P^n \approx Q^n : I^n \to O^n$ explicit, we write the approximate equality judgement as $\Delta^1 \vdash P^1 \approx Q^1 : I^1 \to O^1, \ldots, \Delta^n \vdash P^n \approx Q^n : I^n \to O^n \Rightarrow \Delta \vdash P \approx Q : I \to O$ width k length l.

For the asymptotic equality of IPDL protocols, we assume a finite set $\mathbb{T}_\approx$ of *axiom families* of the form $\{\Delta_\lambda \vdash P_\lambda \approx Q_\lambda : I_\lambda \to O_\lambda\}_{\lambda\in\mathbb{N}}$. In this setting, the asymptotic equivalence of two protocol families $\{\Delta_\lambda \vdash P_\lambda : I_\lambda \to O_\lambda\}_{\lambda\in\mathbb{N}}$ and $\{\Delta_\lambda \vdash Q_\lambda : I_\lambda \to O_\lambda\}_{\lambda\in\mathbb{N}}$ with pointwise-identical typing judgements takes the form of the judgement $\mathbb{T}_\approx \Rightarrow \{\Delta_\lambda \vdash P_\lambda \approx Q_\lambda : I_\lambda \to O_\lambda\}_{\lambda\in\mathbb{N}}$, see Figure 12.

Specifically, for any fixed λ we obtain an approximate theory by selecting from each axiom family in $\mathbb{T}_\approx$ the axiom corresponding to λ. Similarly, from each of the two protocol families we select the protocol corresponding to λ, which gives us two concrete protocols to equate approximately. We recall that an approximate equality judgement is tagged by a pair of parameters k and l. Letting $\lambda \in \mathbb{N}$ vary thus gives us two functions k_λ and l_λ, and we require that these be bounded by a polynomial. We can summarize the asymptotic judgement as saying that the protocol families are pointwise approximately equal, and both the width and length of the derivation, as well as the number of input and output channels are bounded by a polynomial in λ.

Whenever we need to make the underlying exact theory $\mathbb{T}$ explicit, we write the asymptotic equality judgement as $\mathbb{T}; \mathbb{T}_\approx \Rightarrow \{\Delta_\lambda \vdash P_\lambda \approx Q_\lambda : I_\lambda \to O_\lambda\}_{\lambda\in\mathbb{N}}$.

2 Operational Semantics of IPDL

In this section we define an operational semantics for expressions, reactions, and protocols. This semantics will validate the *exact* fragment of our equational logic and prove perfect observational equivalence. To give semantics to user-defined symbols, we define interpretations:

Definition 2 (Interpretation)**.** *An interpretation* $[\![-]\!]$ *for a signature* Σ *associates:*

- *to each type symbol* t *a bitstring length* $[\![\mathsf{t}]\!] \in \mathbb{N}$*;*
- *to each function symbol* $\mathsf{f} : \sigma \to \tau$ *a function* $[\![\mathsf{f}]\!]$ *from bitstrings* $\{0,1\}^{[\![\sigma]\!]}$ *to bitstrings* $\{0,1\}^{[\![\tau]\!]}$*;*
- *to each distribution symbol* $\mathsf{d} : \sigma \to \tau$ *a function* $[\![\mathsf{d}]\!]$ *from bitstrings* $\{0,1\}^{[\![\sigma]\!]}$ *to* distributions *on bitstrings* $\{0,1\}^{[\![\tau]\!]}$.

In the above, we naturally lift the interpretation $[\![-]\!]$ to data types by setting

$$\begin{aligned} [\![1]\!] &:= 0 \\ [\![\mathsf{Bool}]\!] &:= 1 \\ [\![\tau \times \sigma]\!] &:= [\![\tau]\!] + [\![\sigma]\!] \end{aligned}$$

To handle partial computations, we augment the syntax of expressions, reactions, and protocols to contain intermediate bitstring values $v \in \{0,1\}^\star$:

$$\begin{array}{llll} \text{Valued Expressions} & e & ::= & v \mid \ldots \\ \text{Valued Reactions} & R, S & ::= & \mathsf{val}\ v \mid \ldots \\ \text{Valued Protocols} & P & ::= & o := v \mid \ldots \end{array}$$

Given an ambient interpretation $[\![-]\!]$ for the signature Σ, we can type the valued counterpart of IPDL constructs as expected: in addition to the regular typing rules, we have

$$\frac{v \in \{0,1\}^{[\![\tau]\!]}}{\Gamma \vdash v : \tau} \qquad \frac{v \in \{0,1\}^{[\![\tau]\!]}}{\Delta;\ \Gamma \vdash \mathsf{val}\ v : I \to \tau} \qquad \frac{o : \tau \in \Delta \qquad o \notin I \qquad v \in \{0,1\}^{[\![\tau]\!]}}{\Delta \vdash (o := v) : I \to \{o\}}$$

The big-step semantics $e \Downarrow v$ for expressions is straightforward – see Figure 13, where we denote the empty bitstring by $()$ and use $v_1 v_2$ for bitstring concatenation. Pairing is given by the aforementioned bitstring concatenation (rule PAIR), and the projections $\mathsf{fst}_{\sigma\times\tau}$ and $\mathsf{snd}_{\sigma\times\tau}$ unambiguously split the pair according to $[\![\sigma]\!]$ and $[\![\tau]\!]$, respectively (rules FST and SND).

Lemma 1 (Determinism of $\Downarrow$ for expressions)**.** *For any well-typed expression* $\Gamma \vdash e : \tau$ *there exists a unique value* v *such that* $e \Downarrow v$*, and* $v \in \{0,1\}^{[\![\tau]\!]}$.

Reactions have a straightforward small-step semantics of the form $R \to \eta$, where η is a probability distribution over reactions. Figure 14 shows the rules, where we write $1[R]$ for the distribution with unit mass at the reaction R, and freely use a distribution in place of a value (rule SAMP) or a reaction (rule BIND-REACT) to indicate the obvious lifting of the corresponding construct to distributions on reactions. All distributions are implicitly finitely supported. Crucially, there is no semantic rule for stepping the reaction $\mathsf{read}\ c$ – we model communication via semantics for protocols, which substitute all instances of read for values.

We give semantics to protocols via two main small-step rules, see Figure 15, where we analogously write $1[P]$ for the distribution with unit mass at the protocol P, and freely use a distribution in place of a reaction (in rule STEP-REACT) or a protocol (rules STEP-COMP-LEFT, STEP-COMP-RIGHT, and STEP-NEW) to indicate the obvious lifting of the corresponding construct to distributions on protocols.

First we have the *output* relation $P \overset{o := v}{\longmapsto} Q$, which is enabled when the reaction for channel o in P terminates, resulting in value v (rule OUT-VAL). When this happens, the value of o is broadcast through the protocol context enveloping P (rules OUT-COMP-LEFT, OUT-COMP-LEFT, and OUT-NEW), resulting in each $\mathsf{read}\ o$ command in other reactions to be substituted with $\mathsf{val}\ v$. Note that the value of o is not broadcast above the new quantifier when the local channel introduced is equal to o.

Next we have the *internal stepping* relation $P \to \eta$, specified similarly to the small-step relation for reactions. The rule STEP-REACT lifts the stepping relation for R to the stepping relation for $o := R$, while the three rules STEP-COMP-LEFT, STEP-COMP-RIGHT, STEP-NEW simply propagate the stepping relation through parallel composition and the new quantifier. The last rule OUT-NEW-HIDE links the output relation with the stepping relation: whenever P steps to P', resulting in the output $o := v$, we have that $\mathsf{new}\ o : \tau\ \mathsf{in}\ P$ steps with unit mass to $\mathsf{new}\ o : \tau\ \mathsf{in}\ P'$.

The big-step operational semantics for reactions $R \Downarrow \eta$, see Figure 16, performs as many steps as possible in an attempt to compute R, resulting in a distribution η on reactions. A reaction that cannot step any further is *final*. We can syntactically describe final reactions as those that have either yielded a final value or have an unresolved read in the leading position (*i.e.*, are *stuck*).

Similarly, the big-step operational semantics for protocols $P \Downarrow \eta$, see Figure 17, performs as many output and internal steps as possible in an attempt to compute P, resulting in a distribution η on protocols. Analogously to reactions, a protocol that cannot step any further is *final*. We can syntactically describe final protocols as those where every channel, including the internal ones, carries either a final value or a reaction that is stuck.

Note that while the semantics for reactions is sequential, both output and internal step relations for protocols are non-deterministic. Indeed, any two channels in a protocol may output in any order. Ordinarily, this presents a problem for reasoning about cryptography, since non-deterministic choice may present a security leak. However, our language introduces *no* way to exploit this extra non-determinism, essentially due to the read commands in reactions being blocking. This is formalized by a *confluence* result for IPDL:

Lemma 2 (Confluence). *If $\Delta \vdash P : I \to O$, then:*

- *If $P \overset{o := v_1}{\longmapsto} Q_1$ and $P \overset{o := v_2}{\longmapsto} Q_2$, then $v_1 = v_2$ and $Q_1 = Q_2$.*
- *If $P \overset{o_1 := v_1}{\longmapsto} Q_1$ and $P \overset{o_2 := v_2}{\longmapsto} Q_2$ with $o_1 \neq o_2$, then there exists Q such that $Q_1 \overset{o_2 := v_2}{\longmapsto} Q$ and $Q_2 \overset{o_1 := v_1}{\longmapsto} Q$.*
- *If $P \overset{o := v}{\longmapsto} Q$ and $P \to \eta$, then there exists η' such that $\eta \overset{o := v}{\longmapsto} \eta'$ and $Q \to \eta'$.*
- *If $P \to \eta_1$ and $P \to \eta_2$, then either $\eta_1 = \eta_2$ or there exists η such that $\eta_1 \to \eta$ and $\eta_2 \to \eta$.*

In the above lemma, we lift the two protocol stepping relations $\overset{o := v}{\longmapsto}$ and $\to$ to distributions in the natural way.

To guarantee termination of the semantics for reactions, we count the maximum number of steps the reaction would take *provided all reads were resolved*:

$$\begin{aligned}
\|\mathsf{val}\ v\| &:= 0 \\
\|\mathsf{ret}\ e\| &:= 1 \\
\|\mathsf{samp}\ (\mathsf{d}\ e)\| &:= 1 \\
\|\mathsf{read}\ c\| &:= 0 \\
\|\mathsf{if}\ e\ \mathsf{then}\ R_1\ \mathsf{else}\ R_2\| &:= \mathsf{max}\,(\|R_1\|, \|R_2\|) + 1 \\
\|x : \tau \leftarrow R;\ S\| &:= (\|R\| + \|S\|) + 1
\end{aligned}$$

We note that $\|-\|$ for reactions is invariant under substitutions, embeddings, and input assignment. As expected, stepping reduces the number of steps left, guaranteeing termination:

Lemma 3. *If $R \to \sum_i c_i\ 1[R_i]$, $c_i \neq 0$, then $\|R_i\| < \|R\|$.*

Corollary 1 (Determinism of $\Downarrow$ for reactions)**.** *For any well-typed reaction $\Delta; \cdot \vdash R : I \to \tau$ there exists a unique distribution η such that $R \Downarrow \eta$. We will denote η by $R\Downarrow$.*

To guarantee termination of the semantics for protocols, we analogously count the maximum number of steps the protocol would take *provided all reads in reactions were resolved*:

$$\begin{aligned} \|\mathsf{0}\| &:= 0 \\ \|o := v\| &:= 0 \\ \|o := R\| &:= \|R\| + 1 \\ \|P \mid\mid Q\| &:= \|P\| + \|Q\| \\ \|\mathsf{new}\ c : \tau\ \mathsf{in}\ P\| &:= \|P\| \end{aligned}$$

As for reactions, $\|-\|$ for protocols is invariant under embeddings and input assignment, and stepping reduces the number of steps left:

Lemma 4. *If $P \xmapsto{o := v} Q$, then $\|Q\| < \|P\|$, and if $P \to \sum_i c_i\ 1[P_i]$, $c_i \neq 0$, then $\|P_i\| < \|P\|$.*

Together with confluence, termination gives us the desired result:

Corollary 2 (Determinism of $\Downarrow$ for protocols)**.** *For any well-typed protocol $\Delta \vdash P : I \to O$ there exists a unique distribution η such that $P \Downarrow \eta$. We will denote η by $P\Downarrow$.*

3 Soundness of Exact Equality in IPDL

Soundness of equality at the expression level means that if we substitute the same valued expression for each free variable, the resulting closed expressions will compute to the same value:

Definition 3. *An axiom $\Gamma \vdash e_1 = e_2 : \tau$ is* sound *if for any valued substitution $\theta : \cdot \to \Gamma$, we have $\theta^\star(e_1)\Downarrow = \theta^\star(e_2)\Downarrow$.*

The ambient IPDL theory for expressions is said to be sound if each of its axioms is sound. It is straightforward to show that this implies overall soundness:

Lemma 5 (Soundness of equality of expressions)**.** *If the ambient IPDL theory for expressions is sound, then for any equal expressions $\Gamma \vdash e_1 = e_2 : \tau$ and any valued substitution $\theta : \cdot \to \Gamma$, we have that $\theta^\star(e_1)\Downarrow = \theta^\star(e_1)\Downarrow$.*

At the reaction level, two equal reactions should behave in a way that is indistinguishable by an external observer. We formally capture this notion of indistinguishability by a logical relation known as a *bisimulation* – a binary relation on distributions on reactions that satisfies certain closure properties, together with the crucial *valuation property* that allows us to jointly partition two related distributions so that any two corresponding components are again related and have the same *value*: a reaction R is said to have value v if R is of the form $\mathsf{val}\ v$ (otherwise the value is undefined), and we lift this notion to distributions on reactions in the obvious way. At the reaction level, we only require the valuation property for those distributions that are *final*, *i.e.*, no reaction in the support steps.

Definition 4 (Reaction bisimulation)**.** *A* reaction bisimulation $\sim$ *is a binary relation on distributions on reactions $\Delta;\ \cdot \vdash R : I \to \tau$ satisfying the following conditions:*

- Closure under convex combinations*: For any distributions $\eta_1 \sim \varepsilon_1$ and $\eta_2 \sim \varepsilon_2$, and any coefficients $c_1, c_2 > 0$ with $c_1 + c_2 = 1$, we have $c_1\eta_1 + c_2\eta_2 \sim c_1\varepsilon_1 + c_2\varepsilon_2$.*
- Closure under input assignment*: For any distributions $\eta \sim \varepsilon$, input channel $i \in I$ of type τ, and value $v \in \{0,1\}^{[\![\tau]\!]}$, we have $\eta[\mathsf{read}\ i := \mathsf{val}\ v] \sim \varepsilon[\mathsf{read}\ i := \mathsf{val}\ v]$.*

- Closure under computation*: For any distributions $\eta \sim \varepsilon$, we have $\eta\Downarrow \sim \varepsilon\Downarrow$.*
- Valuation property*: For any distributions $\eta \sim \varepsilon$ that are final, there exists a joint convex combination*

$$\eta = \sum_i c_i\, \eta_i \;\sim\; \sum_i c_i\, \varepsilon_i = \varepsilon$$

with $c_i > 0$ and $\sum_i c_i = 1$, such that

  - *the respective components $\eta_i \sim \varepsilon_i$ are again related, and*
  - *the distributions η_i and ε_j have the same value v or lack thereof if and only if $i = j$.*

Crucially, we note that the joint convex combination in the valuation property is unique up to the order of the summands. We now describe one canonical way to construct reaction bisimulations:

Definition 5. *Let $\sim$ be an arbitrary binary relation on distributions on reactions $\Delta;\ \cdot \vdash R : I \to \tau$. The* lifting *$\mathcal{L}(()\sim)$ is the closure of $\sim$ under joint convex combinations. Explicitly, $\mathcal{L}(()\sim)$ is defined by*

$$\sum_i c_i\, \eta_i\ \mathcal{L}(()\sim)\ \sum_i c_i\, \varepsilon_i$$

for coefficients $c_i > 0$ with $\sum_i c_i = 1$ and distributions $\eta_i \sim \varepsilon_i$.

Lemma 6. *Let $\sim$ be a binary relation on distributions on reactions $\Delta;\ \cdot \vdash R : I \to \tau$ with the following properties:*

- Closure under input assignment*: For any distributions $\eta \sim \varepsilon$, input channel $i \in I$ of type τ, and value $v \in \{0,1\}^{[\![\tau]\!]}$, we have $\eta[\mathsf{read}\ i := \mathsf{val}\ v] \sim \varepsilon[\mathsf{read}\ i := \mathsf{val}\ v]$.*
- Lifting closure under computation*: For any distributions $\eta \sim \varepsilon$, we have $\eta\Downarrow\, \mathcal{L}(()\sim)\, \varepsilon\Downarrow$.*
- Valuation property*: For any distributions $\eta \sim \varepsilon$ that are final, there exists a joint convex combination*

$$\eta = \sum_i c_i\, \eta_i \;\sim\; \sum_i c_i\, \varepsilon_i = \varepsilon$$

with $c_i > 0$ and $\sum_i c_i = 1$, such that

  - *the respective components $\eta_i \sim \varepsilon_i$ are again related, and*
  - *the distributions η_i and ε_j have the same value v or lack thereof if and only if $i = j$.*

Then the lifting $\mathcal{L}(()\sim)$ is a reaction bisimulation.

Lemma 7. *We have the following:*

- *The identity relation is a reaction bisimulation.*
- *The inverse of a reaction bisimulation is a reaction bisimulation.*
- *The composition of two reaction bisimulations is a reaction bisimulation.*

Example 1. *Fix two expressions $\cdot \vdash e_1 : \sigma$ and $\cdot \vdash e_2 : \sigma$ such that $e_1\Downarrow = e_2\Downarrow$. Then the relation $\sim$ defined by*

- $1[R(x := e_1)] \sim 1[R(x := e_2)]$ *for reaction $\Delta;\ x : \sigma \vdash R : I \to \tau$*

is a reaction bisimulation.

Having defined reaction bisimulations, we can now formally state what it means for reaction equality to be sound:

Definition 6. *An axiom $\Delta;\ \Gamma \vdash R_1 = R_2 : I \to \tau$ is* sound *if there is a reaction bisimulation $\sim$ such that for any valued substitution $\theta : \cdot \to \Gamma$, we have $1[\theta^\star(R_1)] \sim 1[\theta^\star(R_2)]$.*

The ambient IPDL theory for reactions is said to be sound if each of its axioms is sound. We now show that this implies overall soundness:

Lemma 8 (Soundness of equality of reactions)**.** *If the ambient IPDL theory for reactions is sound, then for any equal reactions* $\Delta;\ \Gamma \vdash R_1 = R_2 : I \to \tau$*, there exists a reaction bisimulation* $\sim$ *such that for any valued substitution* $\theta : \cdot \to \Gamma$*, we have* $1[\theta^\star(R_1)] \sim 1[\theta^\star(R_2)]$.

Proof. We first replace the exchange rule EXCH by the three rules EXCH-SAMP-SAMP, EXCH-SAMP-READ, and EXCH-READ-READ in Figure 18; it is easy to see that this new set of rules is equivalent to the original one. We now proceed by induction on the alternative set of rules for reaction equality. We will freely use a distribution in place of a value (rule EXCH-SAMP-READ) or a reaction (rules EMBED, CONG-BIND) to indicate the obvious lifting of the corresponding construct to distributions on reactions.

- REFL: Our desired bisimulation is the identity relation.
- SYM: Our desired bisimulation is the inverse of the bisimulation obtained from the premise.
- TRANS: Our desired bisimulation is the composition of the two bisimulations obtained from the two premises.
- AXIOM: The desired bisimulation exists by assumption.
- INPUT-UNUSED: Our desired bisimulation is precisely the bisimulation obtained from the premise, seen as a bisimulation on distributions on reactions with the additional input i.
- SUBST: Our desired bisimulation is precisely the bisimulation obtained from the premise.
- EMBED: Let $\sim$ be the bisimulation obtained from the premise. Our desired bisimulation $\sim_\phi$ is defined by
  - $\phi^\star(\eta) \sim_\phi \phi^\star(\varepsilon)$ if $\eta \sim \varepsilon$
- CONG-RET: Our desired bisimulation is the lifting of the relation $\sim$ defined by
  - $1[\mathsf{ret}\ e] \sim 1[\mathsf{ret}\ e']$ for
    - expressions $\cdot \vdash e : \tau$ and $\cdot \vdash e' : \tau$such that $e \Downarrow = e' \Downarrow$
  - $1[\mathsf{val}\ v] \sim 1[\mathsf{val}\ v]$ for value $v \in \{0,1\}^{[\![\tau]\!]}$
- CONG-SAMP: Our desired bisimulation is the lifting of the relation $\sim$ defined by
  - $1[\mathsf{samp}\ (\mathsf{d}\ e)] \sim 1[\mathsf{samp}\ (\mathsf{d}\ e')]$ for
    - expressions $\cdot \vdash e : \tau$ and $\cdot \vdash e' : \tau$such that $e \Downarrow = e' \Downarrow$
  - $1[\mathsf{val}\ v] \sim 1[\mathsf{val}\ v]$ for value $v \in \{0,1\}^{[\![\tau]\!]}$
- CONG-IF: Let $\sim_1$ and $\sim_2$ be the two bisimulations obtained from the two premises. Our desired bisimulation is the lifting of the relation $\sim_{\mathsf{if}}$ defined by
  - $1[\mathsf{if}\ e\ \mathsf{then}\ R_1\ \mathsf{else}\ R_2] \sim_{\mathsf{if}} 1[\mathsf{if}\ e'\ \mathsf{then}\ R_1'\ \mathsf{else}\ R_2']$ for
    - expressions $\cdot \vdash e : \mathsf{Bool}$ and $\cdot \vdash e' : \mathsf{Bool}$such that $e \Downarrow = e' \Downarrow$
    - reactions $\Delta;\ \cdot \vdash R_1 : I \to \tau$ and $\Delta;\ \cdot \vdash R_1' : I \to \tau$ such that $1[R_1] \sim_1 1[R_1']$
    - reactions $\Delta;\ \cdot \vdash R_2 : I \to \tau$ and $\Delta;\ \cdot \vdash R_2' : I \to \tau$ such that $1[R_2] \sim_2 1[R_2']$
  - $\eta_1 \sim_{\mathsf{if}} \eta_1'$ if $\eta_1 \sim_1 \eta_1'$
  - $\eta_2 \sim_{\mathsf{if}} \eta_2'$ if $\eta_2 \sim_2 \eta_2'$
- CONG-BIND: Let $\sim_1$ and $\sim_2$ be the two bisimulations obtained from the two premises. Our desired bisimulation is the lifting of the relation $\sim_{\mathsf{bind}}$ defined by
  - $(x \leftarrow \eta;\ S) \sim_{\mathsf{bind}} (x \leftarrow \eta';\ S')$ for
    - distributions $\eta \sim_1 \eta'$

    - reactions $\Delta;\ x : \sigma \vdash S : I \to \tau$ and $\Delta;\ x : \sigma \vdash S' : I \to \tau$ such that for any value $v \in \{0,1\}^{[\![\sigma]\!]}$, we have $1[S(x := v)] \sim_2 1[S'(x := v)]$
  - $\varepsilon \sim_{\mathsf{bind}} \varepsilon'$ if $\varepsilon \sim_2 \varepsilon'$

- RET-BIND: Our desired bisimulation is the lifting of the relation $\sim$ defined by
  - $1[x \leftarrow \mathsf{ret}\ e;\ R] \sim 1[R(x := e)]$ for expression $\cdot \vdash e : \sigma$ and reaction $\Delta;\ x : \sigma \vdash R : I \to \tau$
  - $1[R(x := v)] \sim 1[R(x := e)]$ for
    - reaction $\Delta;\ x : \sigma \vdash R : I \to \tau$
    - expression $\cdot \vdash e : \sigma$ and value $v \in \{0,1\}^{[\![\sigma]\!]}$ such that $e \Downarrow v$

- BIND-RET: Our desired bisimulation is the lifting of the relation $\sim$ defined by
  - $1[x \leftarrow R;\ \mathsf{ret}\ x] \sim 1[R]$ for reaction $\Delta;\ \cdot \vdash R : I \to \tau$
  - $1[\mathsf{val}\ v] \sim 1[\mathsf{val}\ v]$ for value $v \in \{0,1\}^{[\![\tau]\!]}$

- BIND-BIND: Our desired bisimulation is the lifting of the relation $\sim$ defined by
  - $1[x_2 \leftarrow (x_1 \leftarrow R_1;\ R_2);\ S] \sim 1[x_1 \leftarrow R_1;\ x_2 \leftarrow R_2;\ S]$ for
    - reaction $\Delta;\ \cdot \vdash R_1 : I \to \sigma_1$
    - reaction $\Delta;\ x_1 : \sigma_1 \vdash R_2 : I \to \sigma_2$
    - reaction $\Delta;\ x_2 : \sigma_2 \vdash S : I \to \tau$
  - $1[x_2 \leftarrow R_2;\ S] \sim 1[x_2 \leftarrow R_2;\ S]$ for
    - reaction $\Delta;\ \cdot \vdash R_2 : I \to \sigma_2$
    - reaction $\Delta;\ x_2 : \sigma_2 \vdash S : I \to \tau$
  - $1[S] \sim 1[S]$ for reaction $\Delta;\ \cdot \vdash S : I \to \tau$

- SAMP-PURE: Our desired bisimulation is the lifting of the relation $\sim$ defined by
  - $1[x \leftarrow \mathsf{samp}\ (\mathsf{d}\ e);\ R] \sim 1[R]$ for reaction $\Delta;\ \cdot \vdash R : I \to \tau$
  - $1[R] \sim 1[R]$ for reaction $\Delta;\ \cdot \vdash R : I \to \tau$

- READ-DET: Our desired bisimulation is the lifting of the relation $\sim$ defined by
  - $1[x \leftarrow \mathsf{read}\ i;\ y \leftarrow \mathsf{read}\ i;\ R] \sim 1[x \leftarrow \mathsf{read}\ i;\ R(y := x)]$ for reaction $\Delta;\ x : \sigma, y : \sigma \vdash R : I \to \tau$
  - $1[x \leftarrow \mathsf{val}\ v;\ y \leftarrow \mathsf{val}\ v;\ R] \sim 1[x \leftarrow \mathsf{val}\ v;\ R(y := x)]$ for
    - reaction $\Delta;\ x : \sigma, y : \sigma \vdash R : I \to \tau$
    - value $v \in \{0,1\}^{[\![\sigma]\!]}$
  - $1[R] \sim 1[R]$ for reaction $\Delta;\ \cdot \vdash R : I \to \tau$

- IF-LEFT: Our desired bisimulation is the lifting of the relation $\sim$ defined by
  - $1[\mathsf{if\ true\ then}\ R_1\ \mathsf{else}\ R_2] \sim 1[R_1]$ for reactions $\Delta;\ \cdot \vdash R_1 : I \to \tau$ and $\Delta;\ \cdot \vdash R_2 : I \to \tau$
  - $1[R_1] \sim 1[R_1]$ for reaction $\Delta;\ \cdot \vdash R_1 : I \to \tau$

- IF-RIGHT: Our desired bisimulation is the lifting of the relation $\sim$ defined by
  - $1[\mathsf{if\ false\ then}\ R_1\ \mathsf{else}\ R_2] \sim 1[R_2]$ for reactions $\Delta;\ \cdot \vdash R_1 : I \to \tau$ and $\Delta;\ \cdot \vdash R_2 : I \to \tau$
  - $1[R_2] \sim 1[R_2]$ for reaction $\Delta;\ \cdot \vdash R_2 : I \to \tau$

- IF-EXT: Our desired bisimulation is the lifting of the relation $\sim$ defined by
  - $1[R(x := e)] \sim 1[\mathsf{if}\ e\ \mathsf{then}\ R(x := \mathsf{true})\ \mathsf{else}\ R(x := \mathsf{false})]$ for
    - reaction $\Delta;\ x : \mathsf{Bool} \vdash R : I \to \tau$

    * expression $\cdot \vdash e : \mathsf{Bool}$
  - $1[R(x := e)] \sim 1[R(x := \mathsf{true})]$ for
    * reaction $\Delta;\ x : \mathsf{Bool} \vdash R : I \to \tau$
    * expression $\cdot \vdash e : \mathsf{Bool}$ such that $e \Downarrow = 1$
  - $1[R(x := e)] \sim 1[R(x := \mathsf{false})]$ for
    * reaction $\Delta;\ x : \mathsf{Bool} \vdash R : I \to \tau$
    * expression $\cdot \vdash e : \mathsf{Bool}$ such that $e \Downarrow = 0$

- EXCH-SAMP-SAMP: Our desired bisimulation is the lifting of the relation $\sim$ defined by
  - $1[x_1 \leftarrow \mathsf{samp}\ (\mathsf{d}_1\ e_1);\ x_2 \leftarrow \mathsf{samp}\ (\mathsf{d}_2\ e_2);\ \mathsf{ret}\ (x_1, x_2)] \sim$
    $1[x_2 \leftarrow \mathsf{samp}\ (\mathsf{d}_2\ e_2);\ x_1 \leftarrow \mathsf{samp}\ (\mathsf{d}_1\ e_1);\ \mathsf{ret}\ (x_1, x_2)]$ for
    * expressions $\cdot \vdash e_1 : \sigma_1$ and $\cdot \vdash e_2 : \sigma_2$
  - $1[\mathsf{val}\ v_1 v_2] \sim 1[\mathsf{val}\ v_1 v_2]$ for values $v_1 \in \{0,1\}^{[\![\tau_1]\!]}$ and $v_2 \in \{0,1\}^{[\![\tau_2]\!]}$

- EXCH-SAMP-READ: Our desired bisimulation is the lifting of the relation $\sim$ defined by
  - $1[x_1 \leftarrow \mathsf{samp}\ (\mathsf{d}\ e);\ x_2 \leftarrow \mathsf{read}\ i;\ \mathsf{ret}\ (x_1, x_2)] \sim 1[x_2 \leftarrow \mathsf{read}\ i;\ x_1 \leftarrow \mathsf{samp}\ (\mathsf{d}\ e);\ \mathsf{ret}\ (x_1, x_2)]$ for
    * expression $\cdot \vdash e : \sigma$
  - $1[x_1 \leftarrow \mathsf{samp}\ (\mathsf{d}\ e);\ x_2 \leftarrow \mathsf{val}\ v_2;\ \mathsf{ret}\ (x_1, x_2)] \sim 1[x_2 \leftarrow \mathsf{val}\ v_2;\ x_1 \leftarrow \mathsf{samp}\ (\mathsf{d}\ e);\ \mathsf{ret}\ (x_1, x_2)]$ for
    * expression $\cdot \vdash e : \sigma$
    * value $v_2 \in \{0,1\}^{[\![\tau_2]\!]}$
  - $\big(x_2 \leftarrow \mathsf{read}\ i;\ \mathsf{ret}\ ([\![\mathsf{d}]\!](v), x_2)\big) \sim 1[x_2 \leftarrow \mathsf{read}\ i;\ x_1 \leftarrow \mathsf{samp}\ (\mathsf{d}\ e);\ \mathsf{ret}\ (x_1, x_2)]$ for
    * expression $\cdot \vdash e : \sigma$ and value $v \in \{0,1\}^{[\![\sigma]\!]}$ such that $e \Downarrow v$
  - $\big(x_2 \leftarrow \mathsf{val}\ v_2;\ \mathsf{ret}\ ([\![\mathsf{d}]\!](e\Downarrow), x_2)\big) \sim 1[x_2 \leftarrow \mathsf{val}\ v_2;\ x_1 \leftarrow \mathsf{samp}\ (\mathsf{d}\ e);\ \mathsf{ret}\ (x_1, x_2)]$ for
    * expression $\cdot \vdash e : \sigma$
    * value $v_2 \in \{0,1\}^{[\![\tau_2]\!]}$
  - $1[\mathsf{val}\ v_1 v_2] \sim 1[\mathsf{val}\ v_1 v_2]$ for values $v_1 \in \{0,1\}^{[\![\tau_1]\!]}$ and $v_2 \in \{0,1\}^{[\![\tau_2]\!]}$

- EXCH-READ-READ: Our desired bisimulation is the lifting of the relation $\sim$ defined by
  - $1[x_1 \leftarrow \mathsf{read}\ i_1;\ x_2 \leftarrow \mathsf{read}\ i_2;\ \mathsf{ret}\ (x_1, x_2)] \sim 1[x_2 \leftarrow \mathsf{read}\ i_2;\ x_1 \leftarrow \mathsf{read}\ i_1;\ \mathsf{ret}\ (x_1, x_2)]$
  - $1[x_1 \leftarrow \mathsf{val}\ v_1;\ x_2 \leftarrow \mathsf{read}\ i_2;\ \mathsf{ret}\ (x_1, x_2)] \sim 1[x_2 \leftarrow \mathsf{read}\ i_2;\ x_1 \leftarrow \mathsf{val}\ v_1;\ \mathsf{ret}\ (x_1, x_2)]$ for
    * value $v_1 \in \{0,1\}^{[\![\tau_1]\!]}$
  - $1[x_1 \leftarrow \mathsf{read}\ i_1;\ x_2 \leftarrow \mathsf{val}\ v_2;\ \mathsf{ret}\ (x_1, x_2)] \sim 1[x_2 \leftarrow \mathsf{val}\ v_2;\ x_1 \leftarrow \mathsf{read}\ i_1;\ \mathsf{ret}\ (x_1, x_2)]$ for
    * value $v_2 \in \{0,1\}^{[\![\tau_2]\!]}$
  - $1[x_1 \leftarrow \mathsf{val}\ v_1;\ x_2 \leftarrow \mathsf{val}\ v_2;\ \mathsf{ret}\ (x_1, x_2)] \sim 1[x_2 \leftarrow \mathsf{val}\ v_2;\ x_1 \leftarrow \mathsf{val}\ v_1;\ \mathsf{ret}\ (x_1, x_2)]$ for
    * values $v_1 \in \{0,1\}^{[\![\tau_1]\!]}$ and $v_2 \in \{0,1\}^{[\![\tau_2]\!]}$
  - $1[x_2 \leftarrow \mathsf{read}\ i_2;\ \mathsf{ret}\ (v_1, x_2)] \sim 1[x_2 \leftarrow \mathsf{read}\ i_2;\ x_1 \leftarrow \mathsf{val}\ v_1;\ \mathsf{ret}\ (x_1, x_2)]$ for value $v_1 \in \{0,1\}^{[\![\tau_1]\!]}$
  - $1[x_1 \leftarrow \mathsf{read}\ i_1;\ x_2 \leftarrow \mathsf{val}\ v_2;\ \mathsf{ret}\ (x_1, x_2)] \sim 1[x_1 \leftarrow \mathsf{read}\ i_1;\ \mathsf{ret}\ (x_1, v_2)]$ for value $v_2 \in \{0,1\}^{[\![\tau_2]\!]}$
  - $1[x_2 \leftarrow \mathsf{val}\ v_2;\ \mathsf{ret}\ (v_1, x_2)] \sim 1[x_2 \leftarrow \mathsf{val}\ v_2;\ x_1 \leftarrow \mathsf{val}\ v_1;\ \mathsf{ret}\ (x_1, x_2)]$ for
    * values $v_1 \in \{0,1\}^{[\![\tau_1]\!]}$ and $v_2 \in \{0,1\}^{[\![\tau_2]\!]}$
  - $1[x_1 \leftarrow \mathsf{val}\ v_1;\ x_2 \leftarrow \mathsf{val}\ v_2;\ \mathsf{ret}\ (x_1, x_2)] \sim 1[x_1 \leftarrow \mathsf{val}\ v_1;\ \mathsf{ret}\ (x_1, v_2)]$ for
    * values $v_1 \in \{0,1\}^{[\![\tau_1]\!]}$ and $v_2 \in \{0,1\}^{[\![\tau_2]\!]}$
  - $1[\mathsf{val}\ v_1 v_2] \sim 1[\mathsf{val}\ v_1 v_2]$ for values $v_1 \in \{0,1\}^{[\![\tau_1]\!]}$ and $v_2 \in \{0,1\}^{[\![\tau_2]\!]}$

□

At last we get to the protocol level. A protocol bisimulation is entirely analogous to a reaction bisimulation, except we require the valuation property to hold: *i)* per output channel o, and *ii)* for all distributions (not necessarily final).

Definition 7 (Protocol bisimulation)**.** *A* protocol bisimulation $\sim$ *is a binary relation on distributions on protocols* $\Delta \vdash P : I \to O$ *satisfying the following conditions:*

- Closure under convex combinations*: For any distributions* $\eta_1 \sim \varepsilon_1$ *and* $\eta_2 \sim \varepsilon_2$*, and any coefficients* $c_1, c_2 > 0$ *with* $c_1 + c_2 = 1$*, we have* $c_1\eta_1 + c_2\eta_2 \sim c_1\varepsilon_1 + c_2\varepsilon_2$*.*
- Closure under input assignment*: For any distributions* $\eta \sim \varepsilon$*, input channel* $i \in I$ *of type* τ*, and value* $v \in \{0,1\}^{[\![\tau]\!]}$*, we have* $\eta[\mathsf{read}\ i := \mathsf{val}\ v] \sim \varepsilon[\mathsf{read}\ i := \mathsf{val}\ v]$*.*
- Closure under computation*: For any distributions* $\eta \sim \varepsilon$*, we have* $\eta \Downarrow \sim \varepsilon \Downarrow$*.*
- Valuation property*: For any output channel* $o \in O$*, and any distributions* $\eta \sim \varepsilon$*, there exists a joint convex combination*

$$\eta = \sum_i c_i\, \eta_i \ \sim \ \sum_i c_i\, \varepsilon_i = \varepsilon$$

with $c_i > 0$ *and* $\sum_i c_i = 1$*, such that*

  - *the respective components* $\eta_i \sim \varepsilon_i$ *are again related, and*
  - *the distributions* η_i *and* ε_j *have the same value* v *or lack thereof on* o *if and only if* $i = j$*.*

Just like for reaction bisimulations, the joint sum in the valuation property is unique up to the order of the summands. We have an analogous canonical way of constructing protocol bisimulations:

Definition 8. *Let* $\sim$ *be an arbitrary binary relation on distributions on protocols* $\Delta \vdash P : I \to O$*. The* lifting $\mathcal{L}(() \sim)$ *is the closure of* $\sim$ *under joint convex combinations. Explicitly,* $\mathcal{L}(() \sim)$ *is defined by*

$$\sum_i c_i\, \eta_i \ \mathcal{L}(() \sim) \ \sum_i c_i\, \varepsilon_i$$

for coefficients $c_i > 0$ *with* $\sum_i c_i = 1$ *and distributions* $\eta_i \sim \varepsilon_i$*.*

Lemma 9. *Let* $\sim$ *be a binary relation on distributions on protocols* $\Delta \vdash P : I \to O$ *with the following properties:*

- Closure under input assignment*: For any distributions* $\eta \sim \varepsilon$*, input channel* $i \in I$ *of type* τ*, and value* $v \in \{0,1\}^{[\![\tau]\!]}$*, we have* $\eta[\mathsf{read}\ i := \mathsf{val}\ v] \sim \varepsilon[\mathsf{read}\ i := \mathsf{val}\ v]$*.*
- Lifting closure under computation*: For any distributions* $\eta \sim \varepsilon$*, we have* $\eta \Downarrow \mathcal{L}(() \sim) \varepsilon \Downarrow$*.*
- Valuation property*: For any output channel* $o \in O$*, and any distributions* $\eta \sim \varepsilon$*, there exists a joint convex combination*

$$\eta = \sum_i c_i\, \eta_i \ \sim \ \sum_i c_i\, \varepsilon_i = \varepsilon$$

with $c_i > 0$ *and* $\sum_i c_i = 1$*, such that*

  - *the respective components* $\eta_i \sim \varepsilon_i$ *are again related, and*
  - *the distributions* η_i *and* ε_j *have the same value* v *or lack thereof on* o *if and only if* $i = j$*.*

Then the lifting $\mathcal{L}(() \sim)$ *is a protocol bisimulation.*

Lemma 10. *We have the following:*

- *The identity relation is a protocol bisimulation.*
- *The inverse of a protocol bisimulation is a protocol bisimulation.*
- *The composition of two protocol bisimulations is a protocol bisimulation.*

We can now formally state what it means for exact protocol equality to be sound:

Definition 9. *An axiom* $\Delta \vdash P_1 = P_2 : I \to O$ *is* sound *if there is a protocol bisimulation* $\sim$ *such that* $1[P_1] \sim 1[P_2]$.

The ambient IPDL theory for protocols is said to be sound if each of its axioms is sound. We now show that this implies overall soundness for exact equality:

Lemma 11 (Soundness of exact equality of protocols)**.** *If the ambient IPDL theory for protocols is sound, then for any equal protocols* $\Delta \vdash P_1 = P_2 : I \to O$*, there exists a protocol bisimulation* $\sim$ *such that* $1[P_1] \sim 1[P_2]$.

Proof. We first replace the rules FOLD-IF-LEFT and FOLD-IF-RIGHT by the equivalent formulation in Figure 19. We now proceed by induction on this alternative set of rules for exact protocol equality. We will freely use a measure in place of a reaction (rule CONG-REACT) or a protocol (rules EMBED, ABSORB-LEFT) to indicate the obvious lifting of the corresponding construct to measures on protocols.

- REFL: Our desired bisimulation is the identity relation.
- SYM: Our desired bisimulation is the inverse of the bisimulation obtained from the premise.
- TRANS: Our desired bisimulation is the composition of the two bisimulations obtained from the two premises.
- AXIOM: The desired bisimulation exists by assumption.
- INPUT-UNUSED: Our desired bisimulation is precisely the bisimulation obtained from the premise, seen as a bisimulation on distributions on protocols with the additional input i.
- EMBED: Let $\sim$ be the bisimulation obtained from the premise. Our desired bisimulation $\sim_\phi$ is defined by
  - $\phi^\star(\eta) \sim_\phi \phi^\star(\varepsilon)$ if $\eta \sim \varepsilon$
- CONG-REACT: Let $\sim$ be the reaction bisimulation obtained from the premise. Our desired bisimulation is the lifting of the relation $\sim_{\mathsf{react}}$ defined by
  - $(o := \eta) \sim_{\mathsf{react}} (o := \eta')$ for distributions $\eta \sim \eta'$
  - $1[o := v] \sim_{\mathsf{react}} 1[o := v]$ for value $v \in \{0,1\}^{[\![\tau]\!]}$
- CONG-COMP-LEFT: Let $\sim$ be the bisimulation obtained from the premise. Our desired bisimulation is the lifting of the relation $\sim_{\mathsf{par}}$ defined by
  - $(\eta \mid\mid Q) \sim_{\mathsf{par}} (\eta' \mid\mid Q)$ for $\eta \sim \eta'$ and protocol $\Delta \vdash Q : I \cup O_1 \to O_2$

  The fact that this is indeed a bisimulation requires a fair amount of work; see Lemma **??**.
- CONG-NEW: Let $\sim$ be the bisimulation obtained from the premise. Our desired bisimulation $\sim_{\mathsf{new}}$ is defined by
  - $(\mathsf{new}\ o : \tau\ \mathsf{in}\ \eta) \sim_{\mathsf{new}} (\mathsf{new}\ o : \tau\ \mathsf{in}\ \eta')$ if $\eta \sim \eta'$
- COMP-COMM: Our desired bisimulation is the lifting of the relation $\sim$ defined by
  - $1[P_1 \mid\mid P_2] \sim 1[P_2 \mid\mid P_1]$ for protocols $\Delta \vdash P_1 : I \cup O_2 \to O_1$ and $\Delta \vdash P_2 : I \cup O_1 \to O_2$
- COMP-ASSOC: Our desired bisimulation is the lifting of the relation $\sim$ defined by
  - $1\big[(P_1 \mid\mid P_2) \mid\mid P_3\big] \sim 1\big[P_1 \mid\mid (P_2 \mid\mid P_3)\big]$ for
    - protocol $\Delta \vdash P_1 : I \cup O_2 \cup O_3 \to O_1$
    - protocol $\Delta \vdash P_2 : I \cup O_1 \cup O_3 \to O_2$
    - protocol $\Delta \vdash P_3 : I \cup O_1 \cup O_2 \to O_3$

- NEW-EXCH: The desired bisimulation is the lifting of the relation $\sim$ defined by
  - $1[\mathsf{new}\ o_1 : \tau_1\ \mathsf{in}\ \mathsf{new}\ o_2 : \tau_2\ \mathsf{in}\ P] \sim 1[\mathsf{new}\ o_2 : \tau_2\ \mathsf{in}\ \mathsf{new}\ o_1 : \tau_1\ \mathsf{in}\ P]$ for
    - protocol $\Delta, o_1 : \tau_1, o_2 : \tau_2 \vdash P : I \to O \cup \{o_1, o_2\}$
- COMP-NEW: Our desired bisimulation is the lifting of the relation $\sim$ defined by
  - $1[P \mid\mid (\mathsf{new}\ o : \tau\ \mathsf{in}\ Q)] \sim 1[\mathsf{new}\ o : \tau\ \mathsf{in}\ (P \mid\mid Q)]$ for
    - protocol $\Delta \vdash P : I \cup O_2 \to O_1$
    - protocol $\Delta, o : \tau \vdash Q : I \cup O_1 \to O_2 \cup \{o\}$
- ABSORB-LEFT: Our desired bisimulation is the lifting of the relation $\sim$ defined by
  - $1[P \mid\mid Q] \sim 1[P]$ for protocols $\Delta \vdash P : I \to O$ and $\Delta \vdash Q : I \cup O \to \varnothing$
- DIVERGE: Our desired bisimulation is the lifting of the relation $\sim$ defined by
  - $1[o := x \leftarrow \mathsf{read}\ o;\ R] \sim 1[o := \mathsf{read}\ o]$ for reaction $\Delta;\ \cdot \vdash R : I \cup \{o\} \to \tau$
- FOLD-IF-LEFT: Our desired bisimulation is the lifting of the relation $\sim$ defined by
  - $1[\mathsf{new}\ l : \tau\ \mathsf{in}\ o := x \leftarrow \mathsf{read}\ b;\ \mathsf{if}\ x\ \mathsf{then}\ \mathsf{read}\ l\ \mathsf{else}\ S_2 \mid\mid l := x \leftarrow \mathsf{read}\ b;\ S_1] \sim 1[o := x \leftarrow \mathsf{read}\ b;\ \mathsf{if}\ x\ \mathsf{then}\ S_1\ \mathsf{else}\ S_2]$ for
    - reaction $\Delta;\ \cdot \vdash S_1 : I \cup \{o\} \to \tau$
    - reaction $\Delta;\ \cdot \vdash S_2 : I \cup \{o\} \to \tau$
  - $1[\mathsf{new}\ l : \tau\ \mathsf{in}\ o := x \leftarrow \mathsf{val}\ v;\ \mathsf{if}\ x\ \mathsf{then}\ \mathsf{read}\ l\ \mathsf{else}\ S_2 \mid\mid l := x \leftarrow \mathsf{val}\ v;\ S_1] \sim 1[o := x \leftarrow \mathsf{val}\ v;\ \mathsf{if}\ x\ \mathsf{then}\ S_1\ \mathsf{else}\ S_2]$ for
    - value $v \in \{0, 1\}$
    - reaction $\Delta;\ \cdot \vdash S_1 : I \cup \{o\} \to \tau$
    - reaction $\Delta;\ \cdot \vdash S_2 : I \cup \{o\} \to \tau$
  - $1[\mathsf{new}\ l : \tau\ \mathsf{in}\ o := \mathsf{read}\ l \mid\mid l := S_1] \sim 1[o := S_1]$ for reaction $\Delta;\ \cdot \vdash S_1 : I \cup \{o\} \to \tau$
  - $1[\mathsf{new}\ l : \tau\ \mathsf{in}\ o := S_2 \mid\mid l := S_1] \sim 1[o := S_2]$ for
    - reaction $\Delta;\ \cdot \vdash S_1 : I \cup \{o\} \to \tau$
    - reaction $\Delta;\ \cdot \vdash S_2 : I \cup \{o\} \to \tau$
  - $1[\mathsf{new}\ l : \tau\ \mathsf{in}\ o := v_2 \mid\mid l := S_1] \sim 1[o := v_2]$ for
    - reaction $\Delta;\ \cdot \vdash S_1 : I \cup \{o\} \to \tau$
    - value $v_2 \in \{0, 1\}^{[\![\tau]\!]}$
  - $1[\mathsf{new}\ l : \tau\ \mathsf{in}\ o := S_2 \mid\mid l := v_1] \sim 1[o := S_2]$ for
    - value $v_1 \in \{0, 1\}^{[\![\tau]\!]}$
    - reaction $\Delta;\ \cdot \vdash S_2 : I \cup \{o\} \to \tau$
  - $1[\mathsf{new}\ l : \tau\ \mathsf{in}\ o := v_2 \mid\mid l := v_1] \sim 1[o := v_2]$ for values $v_1, v_2 \in \{0, 1\}^{[\![\tau]\!]}$
- FOLD-IF-RIGHT: Our desired bisimulation is the lifting of the relation $\sim$ defined by
  - $1[\mathsf{new}\ r : \tau\ \mathsf{in}\ o := x \leftarrow \mathsf{read}\ b;\ \mathsf{if}\ x\ \mathsf{then}\ S_1\ \mathsf{else}\ \mathsf{read}\ r \mid\mid r := x \leftarrow \mathsf{read}\ b;\ S_2] \sim 1[o := x \leftarrow \mathsf{read}\ b;\ \mathsf{if}\ x\ \mathsf{then}\ S_1\ \mathsf{else}\ S_2]$ for
    - reaction $\Delta;\ \cdot \vdash S_1 : I \cup \{o\} \to \tau$
    - reaction $\Delta;\ \cdot \vdash S_2 : I \cup \{o\} \to \tau$
  - $1[\mathsf{new}\ r : \tau\ \mathsf{in}\ o := x \leftarrow \mathsf{val}\ v;\ \mathsf{if}\ x\ \mathsf{then}\ S_1\ \mathsf{else}\ \mathsf{read}\ r \mid\mid r := x \leftarrow \mathsf{val}\ v;\ S_2] \sim 1[o := x \leftarrow \mathsf{val}\ v;\ \mathsf{if}\ x\ \mathsf{then}\ S_1\ \mathsf{else}\ S_2]$ for
    - value $v \in \{0, 1\}$

    * reaction $\Delta;\ \cdot \vdash S_1 : I \cup \{o\} \to \tau$
    * reaction $\Delta;\ \cdot \vdash S_2 : I \cup \{o\} \to \tau$
  - $1[\mathsf{new}\ r : \tau\ \mathsf{in}\ o := \mathsf{read}\ r \mid\mid r := S_2] \sim 1[o := S_2]$ for reaction $\Delta;\ \cdot \vdash S_2 : I \cup \{o\} \to \tau$
  - $1[\mathsf{new}\ r : \tau\ \mathsf{in}\ o := S_1 \mid\mid r := S_2] \sim 1[o := S_1]$ for
    * reaction $\Delta;\ \cdot \vdash S_1 : I \cup \{o\} \to \tau$
    * reaction $\Delta;\ \cdot \vdash S_2 : I \cup \{o\} \to \tau$
  - $1[\mathsf{new}\ r : \tau\ \mathsf{in}\ o := v_1 \mid\mid r := S_2] \sim 1[o := v_1]$ for
    * value $v_1 \in \{0,1\}^{[\![\tau]\!]}$
    * reaction $\Delta;\ \cdot \vdash S_2 : I \cup \{o\} \to \tau$
  - $1[\mathsf{new}\ r : \tau\ \mathsf{in}\ o := S_1 \mid\mid r := v_2] \sim 1[o := S_1]$ for
    * reaction $\Delta;\ \cdot \vdash S_1 : I \cup \{o\} \to \tau$
    * value $v_2 \in \{0,1\}^{[\![\tau]\!]}$
  - $1[\mathsf{new}\ r : \tau\ \mathsf{in}\ o := v_1 \mid\mid r := v_2] \sim 1[o := v_1]$ for values $v_1, v_2 \in \{0,1\}^{[\![\tau]\!]}$

- FOLD-BIND: Our desired bisimulation is the lifting of the relation $\sim$ defined by
  - $1[\mathsf{new}\ c : \tau_1\ \mathsf{in}\ o := x \leftarrow \mathsf{read}\ c;\ R_2 \mid\mid c := R_1] \sim 1[o := x \leftarrow R_1;\ R_2]$ for
    * reaction $\Delta;\ \cdot \vdash R_1 : I \cup \{o\} \to \tau_1$
    * reaction $\Delta;\ x : \tau_1 \vdash R_2 : I \cup \{o\} \to \tau_2$
  - $1[\mathsf{new}\ c : \tau_1\ \mathsf{in}\ o := R_2 \mid\mid c := v_1] \sim 1[o := R_2]$ for
    * value $v_1 \in \{0,1\}^{[\![\tau_1]\!]}$
    * reaction $\Delta;\ \cdot \vdash R_2 : I \cup \{o\} \to \tau_2$
  - $1[\mathsf{new}\ c : \tau_1\ \mathsf{in}\ o := v_2 \mid\mid c := v_1] \sim 1[o := v_2]$ for
    * values $v_1 \in \{0,1\}^{[\![\tau_1]\!]}$ and $v_2 \in \{0,1\}^{[\![\tau_2]\!]}$

- SUBSUME: Our desired bisimulation is the lifting of the relation $\sim$ defined by
  - $1[o_1 := x_0 \leftarrow \mathsf{read}\ o_0;\ R_1 \mid\mid o_2 := x_0 \leftarrow \mathsf{read}\ o_0;\ x_1 \leftarrow \mathsf{read}\ o_1;\ R_2] \sim$
    $1[o_1 := x_0 \leftarrow \mathsf{read}\ o_0;\ R_1 \mid\mid o_2 := x_1 \leftarrow \mathsf{read}\ o_1;\ R_2]$ for
    * reaction $\Delta;\ x_0 : \tau_0 \vdash R_1 : I \cup \{o_1, o_2\} \to \tau_1$
    * reaction $\Delta;\ x_1 : \tau_1 \vdash R_2 : I \cup \{o_1, o_2\} \to \tau_2$
  - $1[o_1 := x_0 \leftarrow \mathsf{val}\ v_0;\ R_1 \mid\mid o_2 := x_0 \leftarrow \mathsf{val}\ v_0;\ x_1 \leftarrow \mathsf{read}\ o_1;\ R_2] \sim$
    $1[o_1 := x_0 \leftarrow \mathsf{val}\ v_0;\ R_1 \mid\mid o_2 := x_1 \leftarrow \mathsf{read}\ o_1;\ R_2]$ for
    * value $v_0 \in \{0,1\}^{[\![\tau_0]\!]}$
    * reaction $\Delta;\ x_0 : \tau_0 \vdash R_1 : I \cup \{o_1, o_2\} \to \tau_1$
    * reaction $\Delta;\ x_1 : \tau_1 \vdash R_2 : I \cup \{o_1, o_2\} \to \tau_2$
  - $1[o_1 := R_1 \mid\mid o_2 := x_1 \leftarrow \mathsf{read}\ o_1;\ R_2] \sim 1[o_1 := R_1 \mid\mid o_2 := x_1 \leftarrow \mathsf{read}\ o_1;\ R_2]$ for
    * reaction $\Delta;\ \cdot \vdash R_1 : I \cup \{o_1, o_2\} \to \tau_1$
    * reaction $\Delta;\ x_1 : \tau_1 \vdash R_2 : I \cup \{o_1, o_2\} \to \tau_2$
  - $1[o_1 := v_1 \mid\mid o_2 := R_2] \sim 1[o_1 := v_1 \mid\mid o_2 := R_2]$ for
    * value $v_1 \in \{0,1\}^{[\![\tau_1]\!]}$
    * reaction $\Delta;\ \cdot \vdash R_2 : I \cup \{o_1, o_2\} \to \tau_2$
  - $1[o_1 := v_1 \mid\mid o_2 := v_2] \sim 1[o_1 := v_1 \mid\mid o_2 := v_2]$ for values $v_1 \in \{0,1\}^{[\![\tau_1]\!]}$ and $v_2 \in \{0,1\}^{[\![\tau_2]\!]}$

- SUBST: Let $\sim$ be the reaction bisimulation obtained from the premise. Our desired bisimulation is the lifting of the relation $\sim_{\mathsf{subst}}$ defined by
  - $\big(o_1 := \eta \mid\mid o_2 := x_1 \leftarrow \mathsf{read}\ o_1;\ R_2\big) \sim_{\mathsf{subst}} \big(o_1 := \eta \mid\mid o_2 := x_1 \leftarrow \eta;\ R_2\big)$ for

* distribution η on reactions Δ; $\cdot \vdash R_1 : I \cup \{o_1, o_2\} \to \tau_1$
* reaction Δ; $\cdot \vdash R_1 : I \cup \{o_1, o_2\} \to \tau_1$ evaluating to the same distribution as η
* reaction Δ; $x_1 : \tau_1 \vdash R_2 : I \cup \{o_1, o_2\} \to \tau_2$

such that $1[x_1 \leftarrow R_1;\ x_1' \leftarrow R_1;\ \mathsf{ret}\ (x_1, x_1')] \sim 1[x_1 \leftarrow R_1;\ \mathsf{ret}\ (x_1, x_1)]$

– $1[o_1 := v_1 \mid\mid o_2 := R_2] \sim_{\mathsf{subst}} 1[o_1 := v_1 \mid\mid o_2 := R_2]$ for
* value $v_1 \in \{0,1\}^{[\![\tau_1]\!]}$
* reaction Δ; $\cdot \vdash R_2 : I \cup \{o_1, o_2\} \to \tau_2$

– $1[o_1 := v_1 \mid\mid o_2 := v_2] \sim_{\mathsf{subst}} 1[o_1 := v_1 \mid\mid o_2 := v_2]$ for values $v_1 \in \{0,1\}^{[\![\tau_1]\!]}$ and $v_2 \in \{0,1\}^{[\![\tau_2]\!]}$

- DROP: Let $\sim$ be the reaction bisimulation obtained from the premise. Our desired bisimulation is the lifting of the relation $\sim_{\mathsf{drop}}$ defined by

– $\left(o_1 := \eta_1 \mid\mid o_2 := x_1 \leftarrow \mathsf{read}\ o_1;\ R_2\right) \sim_{\mathsf{drop}} \left(o_1 := \eta_1 \mid\mid o_2 := \eta_2\right)$ for
* measure η_1 on reactions Δ; $\cdot \vdash R_1 : I \cup \{o_1, o_2\} \to \tau_1$
* reaction Δ; $\cdot \vdash R_1 : I \cup \{o_1, o_2\} \to \tau_1$ such that
  i) R_1 either evaluates to the same distribution as η_1, or
  ii) there exists a measure $\overline{\eta_1}$ on reactions Δ; $\cdot \vdash R_1 : I \cup \{o_1, o_2\} \to \tau_1$ such that R_1 evaluates to the same distribution as $\eta_1 + \overline{\eta_1}$
* distribution η_2 on reactions Δ; $\cdot \vdash R_2 : I \cup \{o_1, o_2\} \to \tau_2$
* reaction Δ; $\cdot \vdash R_2 : I \cup \{o_1, o_2\} \to \tau_2$ evaluating to the same distribution as η_2

such that $1[x_1 \leftarrow R_1;\ R_2] \sim 1[R_2]$

– $(o_1 := v_1 \mid\mid o_2 := R_2) \sim_{\mathsf{drop}} (o_1 := v_1 \mid\mid o_2 := R_2)$ for
* value $v_1 \in \{0,1\}^{[\![\tau_1]\!]}$
* reaction Δ; $\cdot \vdash R_2 : I \cup \{o_1, o_2\} \to \tau_2$

– $(o_1 := v_1 \mid\mid o_2 := v_2) \sim_{\mathsf{drop}} (o_1 := v_1 \mid\mid o_2 := v_2)$ for values $v_1 \in \{0,1\}^{[\![\tau_1]\!]}$ and $v_2 \in \{0,1\}^{[\![\tau_2]\!]}$

□

The remainder of this section is devoted to proving the following lemma:

Lemma 12 (Compositionality for the exact equality of protocols)**. ??** *Let $\sim$ be a bisimulation on protocols $\Delta \vdash P : I \cup O_2 \to O_1$. Then the lifting of the relation $\sim_{\mathsf{par}}$ defined by*

- $(\eta \mid\mid Q) \sim_{\mathsf{par}} (\eta' \mid\mid Q)$ *for $\eta \sim \eta'$ and protocol $\Delta \vdash Q : I \cup O_1 \to O_2$*

is a protocol bisimulation.

Proof. The one property difficult to verify is lifting closure under computation: for any protocol $\Delta \vdash Q : I \cup O_1 \to O_2$, and any distributions $\eta \sim \eta'$, we have $(\eta \mid\mid Q) \Downarrow \mathcal{L}((\,) \sim_{\mathsf{par}})(\eta' \mid\mid Q) \Downarrow$. The difficulty arises from the *global* nature of the protocol semantics: in the composition $P \mid\mid Q$, a step of the form $P \xrightarrow{o := v} P'$ *changes* the protocol Q (specifically to $Q[\mathsf{read}\ o := \mathsf{val}\ v]$). This makes it hard to express the computation of $P \mid\mid Q$ in terms of the computation of P, because in the course of the latter we are simultaneously probabilistically updating Q. We now have all the preliminaries necessary to prove that $\sim_{\mathsf{par}}$ enjoys lifting closure under computation.

Since the set O_1 of outputs is finite, we can apply the valuation property of $\sim$ in succession for each output channel $o \in O_1$, until we end up with the special case when η and η' have the same value v or lack thereof on each output channel. In other words, it suffices to prove the following: □

$$\boxed{\Delta \vdash P = Q : I \to O}$$

$$\frac{\Delta \vdash P : I \to O}{\Delta \vdash P = P : I \to O}\ \textsc{refl} \qquad \frac{\Delta \vdash P_1 = P_2 : I \to O}{\Delta \vdash P_2 = P_1 : I \to O}\ \textsc{sym}$$

$$\frac{\Delta \vdash P_1 = P_2 : I \to O \qquad \Delta \vdash P_2 = P_3 : I \to O}{\Delta \vdash P_1 = P_3 : I \to O}\ \textsc{trans} \qquad \frac{\Delta \vdash P = Q : I \to O\ \mathsf{axiom}}{\Delta \vdash P = Q : I \to O}\ \textsc{axiom}$$

$$\frac{\phi : \Delta_1 \to \Delta_2 \qquad \Delta_2 \vdash P = Q : I \to O}{\Delta_1 \vdash \phi^\star(P) = \phi^\star(Q) : \phi^\star(I) \to \phi^\star(O)}\ \textsc{embed} \qquad \frac{i \notin I \cup O \qquad \Delta \vdash P_1 = P_2 : I \to O}{\Delta \vdash P_1 = P_2 : I \cup \{i\} \to O}\ \textsc{input-unused}$$

$$\frac{o : \tau \in \Delta \qquad o \notin I \qquad \Delta;\ \cdot \vdash R = R' : I \cup \{o\} \to \tau}{\Delta \vdash (o := R) = (o := R') : I \to \{o\}}\ \textsc{cong-react}$$

$$\frac{\Delta \vdash P = P' : I \cup O_2 \to O_1 \qquad \Delta \vdash Q : I \cup O_1 \to O_2}{\Delta \vdash P \mid\mid Q = P' \mid\mid Q : I \to O_1 \cup O_2}\ \textsc{cong-comp-left}$$

$$\frac{\Delta, o : \tau \vdash P = P' : I \to O \cup \{o\}}{\Delta \vdash (\mathsf{new}\ o : \tau\ \mathsf{in}\ P) = (\mathsf{new}\ o : \tau\ \mathsf{in}\ P') : I \to O}\ \textsc{cong-new}$$

$$\frac{\Delta \vdash P_1 : I \cup O_2 \to O_1 \qquad \Delta \vdash P_2 : I \cup O_1 \to O_2}{\Delta \vdash P_1 \mid\mid P_2 = P_2 \mid\mid P_1 : I \to O_1 \cup O_2}\ \textsc{comp-comm}$$

$$\frac{\Delta \vdash P_1 : I \cup O_2 \cup O_3 \to O_1 \qquad \Delta \vdash P_2 : I \cup O_1 \cup O_3 \to O_2 \qquad \Delta \vdash P_3 : I \cup O_1 \cup O_2 \to O_3}{\Delta \vdash (P_1 \mid\mid P_2) \mid\mid P_3 = P_1 \mid\mid (P_2 \mid\mid P_3) : I \to O_1 \cup O_2 \cup O_3}\ \textsc{comp-assoc}$$

$$\frac{\Delta, o_1 : \tau_1, o_2 : \tau_2 \vdash P : I \to O \cup \{o_1, o_2\}}{\Delta \vdash \big(\mathsf{new}\ o_1 : \tau_1\ \mathsf{in}\ \mathsf{new}\ o_2 : \tau_2\ \mathsf{in}\ P\big) = \big(\mathsf{new}\ o_2 : \tau_2\ \mathsf{in}\ \mathsf{new}\ o_1 : \tau_1\ \mathsf{in}\ P\big) : I \to O}\ \textsc{new-exch}$$

$$\frac{\Delta \vdash P : I \cup O_2 \to O_1 \qquad \Delta, o : \tau \vdash Q : I \cup O_1 \to O_2 \cup \{o\}}{\Delta \vdash P \mid\mid (\mathsf{new}\ o : \tau\ \mathsf{in}\ Q) = \mathsf{new}\ o : \tau\ \mathsf{in}\ (P \mid\mid Q) : I \to O_1 \cup O_2}\ \textsc{comp-new}$$

$$\frac{\Delta \vdash P : I \to O \qquad \Delta \vdash Q : I \cup O \to \varnothing}{\Delta \vdash P \mid\mid Q = P : I \to O}\ \textsc{absorb-left}$$

Figure 9: Exact equality for IPDL protocols. Additional rules are given in Figure 10.

$$\boxed{\Delta \vdash P = Q : I \to O}$$

$$\frac{o : \tau \in \Delta \qquad o \notin I \qquad \Delta;\ \cdot \vdash R : I \cup \{o\} \to \tau}{\Delta \vdash (o := x : \tau \leftarrow \mathsf{read}\ o;\ R) = (o := \mathsf{read}\ o) : I \to \{o\}}\ \textsc{diverge}$$

$$\frac{o \notin I \qquad b \in I \qquad b : \mathsf{Bool}, o : \tau \in \Delta \qquad \Delta;\ \cdot \vdash S_1 : I \cup \{o\} \to \tau \qquad \Delta;\ \cdot \vdash S_2 : I \cup \{o\} \to \tau}{\begin{array}{c}\Delta \vdash \big(\mathsf{new}\ l : \tau\ \mathsf{in}\ o := x : \mathsf{Bool} \leftarrow \mathsf{read}\ b;\ \mathsf{if}\ x\ \mathsf{then}\ \mathsf{read}\ l\ \mathsf{else}\ S_2 \mid\mid l := S_1\big) = \\ \big(o := x : \mathsf{Bool} \leftarrow \mathsf{read}\ b;\ \mathsf{if}\ x\ \mathsf{then}\ S_1\ \mathsf{else}\ S_2\big) : I \to \{o\}\end{array}}\ \textsc{fold-if-left}$$

$$\frac{o \notin I \qquad b \in I \qquad b : \mathsf{Bool}, o : \tau \in \Delta \qquad \Delta;\ \cdot \vdash S_1 : I \cup \{o\} \to \tau \qquad \Delta;\ \cdot \vdash S_2 : I \cup \{o\} \to \tau}{\begin{array}{c}\Delta \vdash \big(\mathsf{new}\ r : \tau\ \mathsf{in}\ o := x : \mathsf{Bool} \leftarrow \mathsf{read}\ b;\ \mathsf{if}\ x\ \mathsf{then}\ S_1\ \mathsf{else}\ \mathsf{read}\ r \mid\mid r := S_2\big) = \\ \big(o := x : \mathsf{Bool} \leftarrow \mathsf{read}\ b;\ \mathsf{if}\ x\ \mathsf{then}\ S_1\ \mathsf{else}\ S_2\big) : I \to \{o\}\end{array}}\ \textsc{fold-if-right}$$

$$\frac{o \notin I \qquad o : \tau_2 \in \Delta \qquad \Delta;\ \cdot \vdash R_1 : I \cup \{o\} \to \tau_1 \qquad \Delta;\ x : \tau_1 \vdash R_2 : I \cup \{o\} \to \tau_2}{\Delta \vdash \big(\mathsf{new}\ c : \tau_1\ \mathsf{in}\ o := x : \tau_1 \leftarrow \mathsf{read}\ c;\ R_2 \mid\mid c := R_1\big) = (o := x : \tau_1 \leftarrow R_1;\ R_2) : I \to \{o\}}\ \textsc{fold-bind}$$

$$\frac{\begin{array}{c}o_1 \neq o_2 \qquad o_1, o_2 \notin I \qquad o_0 \in I \cup \{o_1, o_2\} \\ o_0 : \tau_0, o_1 : \tau_1, o_2 : \tau_2 \in \Delta \qquad \Delta;\ x_0 : \tau_0 \vdash R_1 : I \cup \{o_1, o_2\} \to \tau_1 \qquad \Delta;\ x_1 : \tau_1 \vdash R_2 : I \cup \{o_1, o_2\} \to \tau_2\end{array}}{\begin{array}{c}\Delta \vdash \big(o_1 := x_0 : \tau_0 \leftarrow \mathsf{read}\ o_0;\ R_1 \mid\mid o_2 := x_0 : \tau_0 \leftarrow \mathsf{read}\ o_0;\ x_1 : \tau_1 \leftarrow \mathsf{read}\ o_1;\ R_2\big) = \\ \big(o_1 := x_0 : \tau_0 \leftarrow \mathsf{read}\ o_0;\ R_1 \mid\mid o_2 := x_1 : \tau_1 \leftarrow \mathsf{read}\ o_1;\ R_2\big) : I \to \{o_1, o_2\}\end{array}}\ \textsc{subsume}$$

$$\frac{\begin{array}{c}o_1 \neq o_2 \qquad o_1, o_2 \notin I \qquad o_1 : \tau_1, o_2 : \tau_2 \in \Delta \\ \Delta;\ \cdot \vdash R_1 : I \cup \{o_1, o_2\} \to \tau_1 \qquad \Delta;\ x_1 : \tau_1 \vdash R_2 : I \cup \{o_1, o_2\} \to \tau_2 \\ \Delta;\ \cdot \vdash \big(x_1 : \tau_1 \leftarrow R_1;\ x_1' : \tau_1 \leftarrow R_1;\ \mathsf{ret}\ (x_1, x_1')\big) = \big(x_1 : \tau_1 \leftarrow R_1;\ \mathsf{ret}\ (x_1, x_1)\big) : I \cup \{o_1, o_2\} \to \tau_1 \times \tau_1\end{array}}{\Delta \vdash \big(o_1 := R_1 \mid\mid o_2 := x_1 : \tau_1 \leftarrow \mathsf{read}\ o_1;\ R_2\big) = \big(o_1 := R_1 \mid\mid o_2 := x_1 : \tau_1 \leftarrow R_1;\ R_2\big) : I \to \{o_1, o_2\}}\ \textsc{subst}$$

$$\frac{\begin{array}{c}o_1 \neq o_2 \qquad o_1, o_2 \notin I \qquad o_1 : \tau_1, o_2 : \tau_2 \in \Delta \qquad \Delta;\ \cdot \vdash R_1 : I \cup \{o_1, o_2\} \to \tau_1 \\ \Delta;\ \cdot \vdash R_2 : I \cup \{o_1, o_2\} \to \tau_2 \qquad \Delta;\ \cdot \vdash (x_1 : \tau_1 \leftarrow R_1;\ R_2) = R_2 : I \cup \{o_1, o_2\} \to \tau_2\end{array}}{\Delta \vdash \big(o_1 := R_1 \mid\mid o_2 := x_1 \leftarrow \mathsf{read}\ o_1;\ R_2\big) = (o_1 := R_1 \mid\mid o_2 := R_2) : I \to \{o_1, o_2\}}\ \textsc{drop}$$

Figure 10: Additional rules for exact equality of IPDL protocols. Distinguishing changes of equalities are highlighed in red.

$$\boxed{\Delta \vdash P \approx Q : I \to O \text{ width } k \text{ length } l}$$

$$\frac{\Delta \vdash P = Q : I \to O}{\Delta \vdash P \approx Q : I \to O \text{ width } 0 \text{ length } 0}\ \text{STRICT} \qquad \frac{\Delta \vdash P_1 \approx P_2 : I \to O \text{ width } k \text{ length } l}{\Delta \vdash P_2 \approx P_1 : I \to O \text{ width } k \text{ length } l}\ \text{SYM}$$

$$\frac{\Delta \vdash P_1 \approx P_2 : I \to O \text{ width } k_1 \text{ length } l_1 \qquad \Delta \vdash P_2 \approx P_3 : I \to O \text{ width } k_2 \text{ length } l_2}{\Delta \vdash P_1 \approx P_3 : I \to O \text{ width } k_1 + k_2 \text{ length } \max(l_1, l_2)}\ \text{TRANS}$$

$$\frac{\Delta \vdash P \approx Q : I \to O \text{ axiom}}{\Delta \vdash P \approx Q : I \to O \text{ width } 1 \text{ length } 0}\ \text{AXIOM}$$

$$\frac{\Delta \vdash P \approx Q : I \to O \text{ width } k_1 \text{ length } l_1 \qquad k_1 \leqslant k_2 \qquad l_1 \leqslant l_2}{\Delta \vdash P \approx Q : I \to O \text{ width } k_2 \text{ length } l_2}\ \text{SUBSUME}$$

$$\frac{i \notin I \cup O \qquad \Delta \vdash P \approx Q : I \to O \text{ width } k \text{ length } l}{\Delta \vdash P \approx Q : I \cup \{i\} \to O \text{ width } k \text{ length } l + 1}\ \text{INPUT-UNUSED}$$

$$\frac{\theta : \Delta_1 \to \Delta_2 \qquad \Delta_1 \vdash P \approx Q : I \to O \text{ width } k \text{ length } l}{\Delta_2 \vdash \theta^\star(P) \approx \theta^\star(Q) : \theta^\star(I) \to \theta^\star(O) \text{ width } k \text{ length } l}\ \text{EMBED}$$

$$\frac{\Delta \vdash P \approx P' : I \cup O_2 \to O_1 \text{ width } k \text{ length } l \qquad \Delta \vdash_\Sigma Q : I \cup O_1 \to O_2}{\Delta \vdash P \mid\mid Q \approx P' \mid\mid Q : I \to O_1 \cup O_2 \text{ width } k \text{ length } l + |Q|}\ \text{CONG-COMP-LEFT}$$

$$\frac{\Delta, o : A \vdash P \approx P' : I \to O \cup \{o\} \text{ width } k \text{ length } l}{\Delta \vdash (\text{new } o : A \text{ in } P) \approx (\text{new } o : A \text{ in } P') : I \to O \text{ width } k \text{ length } l}\ \text{CONG-NEW}$$

Figure 11: Approximate equality for IPDL protocols.

$$\boxed{\{\Delta_\lambda^1 \vdash P_\lambda^1 \approx Q_\lambda^1 : I_\lambda^1 \to O_\lambda^1\}_{\lambda \in \mathbb{N}}, \ldots, \{\Delta_\lambda^n \vdash P_\lambda^n \approx Q_\lambda^n : I_\lambda^n \to O_\lambda^n\}_{\lambda \in \mathbb{N}} \Rightarrow \{\Delta_\lambda \vdash P_\lambda \approx Q_\lambda : I_\lambda \to O_\lambda\}_{\lambda \in \mathbb{N}}}$$

$$\frac{\begin{array}{c}\forall \lambda, \Delta_\lambda^1 \vdash P_\lambda^1 \approx Q_\lambda^1 : I_\lambda^1 \to O_\lambda^1, \ldots, \Delta_\lambda^n \vdash P_\lambda^n \approx Q_\lambda^n : I_\lambda^n \to O_\lambda^n \Rightarrow \Delta_\lambda \vdash P_\lambda \approx Q_\lambda : I_\lambda \to O_\lambda \text{ width } k_\lambda \text{ length } l_\lambda \\ k_\lambda = \mathsf{O}(\mathsf{poly}(\lambda)) \qquad l_\lambda = \mathsf{O}(\mathsf{poly}(\lambda)) \qquad |I_\lambda| = \mathsf{O}(\mathsf{poly}(\lambda)) \qquad |O_\lambda| = \mathsf{O}(\mathsf{poly}(\lambda))\end{array}}{\{\Delta_\lambda^1 \vdash P_\lambda^1 \approx Q_\lambda^1 : I_\lambda^1 \to O_\lambda^1\}_{\lambda \in \mathbb{N}}, \ldots, \{\Delta_\lambda^n \vdash P_\lambda^n \approx Q_\lambda^n : I_\lambda^n \to O_\lambda^n\}_{\lambda \in \mathbb{N}} \Rightarrow \{\Delta_\lambda \vdash P_\lambda \approx Q_\lambda : I_\lambda \to O_\lambda\}_{\lambda \in \mathbb{N}}}$$

Figure 12: Asymptotic equivalence for IPDL protocol families.

$$\boxed{e \Downarrow v}$$

$$\frac{}{v \Downarrow v}\ \text{VAL} \qquad \frac{}{\checkmark \Downarrow ()}\ \text{CHECK} \qquad \frac{}{\mathsf{true} \Downarrow 1}\ \text{TRUE} \qquad \frac{}{\mathsf{false} \Downarrow 0}\ \text{FALSE} \qquad \frac{e \Downarrow v}{(\mathsf{f}\ e) \Downarrow [\![\mathsf{f}]\!](v)}\ \text{APP}$$

$$\frac{e_1 \Downarrow v_1 \qquad e_2 \Downarrow v_2}{(e_1, e_2) \Downarrow v_1 v_2}\ \text{PAIR} \qquad \frac{e \Downarrow v_1 v_2 \qquad v_1 \in \{0,1\}^{[\![\sigma]\!]}}{(\mathsf{fst}_{\sigma \times \tau}\ e) \Downarrow v_1}\ \text{FST} \qquad \frac{e \Downarrow v_1 v_2 \qquad v_2 \in \{0,1\}^{[\![\tau]\!]}}{(\mathsf{snd}_{\sigma \times \tau})\ e \Downarrow v_2}\ \text{SND}$$

Figure 13: Big-step operational semantics for IPDL expressions.

$$\boxed{R \to \eta}$$

$$\frac{e \Downarrow v}{\mathsf{ret}\ e \to 1[\mathsf{val}\ v]}\ \textsc{ret} \qquad \frac{e \Downarrow v}{\mathsf{samp}\ (\mathsf{d}\ e) \to \mathsf{val}\ [\![\mathsf{d}]\!](v)}\ \textsc{samp} \qquad \frac{e \Downarrow 1}{(\mathsf{if}\ e\ \mathsf{then}\ R_1\ \mathsf{else}\ R_2) \to 1[R_1]}\ \textsc{if-true}$$

$$\frac{e \Downarrow 0}{(\mathsf{if}\ e\ \mathsf{then}\ R_1\ \mathsf{else}\ R_2) \to 1[R_2]}\ \textsc{if-false} \qquad \frac{}{(x : \sigma \leftarrow \mathsf{val}\ v;\ S) \to 1\big[S[x := v]\big]}\ \textsc{bind-val}$$

$$\frac{R \to \eta}{(x : \sigma \leftarrow R;\ S) \to (x : \sigma \leftarrow \eta;\ S)}\ \textsc{bind-react}$$

Figure 14: Small-step operational semantics for IPDL reactions.

$$\boxed{P \xmapsto{o := v} Q}$$

$$\frac{}{(o := \mathsf{val}\ v) \xmapsto{o := v} (o := v)}\ \textsc{out-react} \qquad \frac{P \xmapsto{o := v} P'}{(P \mid\mid Q) \xmapsto{o := v} \big(P' \mid\mid Q[\mathsf{read}\ o := \mathsf{val}\ v]\big)}\ \textsc{out-comp-left}$$

$$\frac{Q \xmapsto{o := v} Q'}{(P \mid\mid Q) \xmapsto{o := v} \big(P[\mathsf{read}\ o := \mathsf{val}\ v] \mid\mid Q'\big)}\ \textsc{out-comp-right} \qquad \frac{P \xmapsto{o := v} P' \qquad o \neq c}{(\mathsf{new}\ c : \tau\ \mathsf{in}\ P) \xmapsto{o := v} (\mathsf{new}\ c : \tau\ \mathsf{in}\ P')}\ \textsc{out-new}$$

$$\boxed{P \to \eta}$$

$$\frac{R \to \eta}{(o := R) \to (o := \eta)}\ \textsc{step-react} \qquad \frac{P \to \eta}{(P \mid\mid Q) \to (\eta \mid\mid Q)}\ \textsc{step-comp-left}$$

$$\frac{Q \to \eta}{(P \mid\mid Q) \to (P \mid\mid \eta)}\ \textsc{step-comp-right} \qquad \frac{P \to \eta}{(\mathsf{new}\ c : \tau\ \mathsf{in}\ P) \to (\mathsf{new}\ c : \tau\ \mathsf{in}\ \eta)}\ \textsc{step-new}$$

$$\frac{P \xmapsto{c := v} P'}{(\mathsf{new}\ c : \tau\ \mathsf{in}\ P) \to 1[\mathsf{new}\ c : \tau\ \mathsf{in}\ P']}\ \textsc{out-new-hide}$$

Figure 15: Small-step operational semantics for IPDL protocols.

$$\boxed{R \text{ stuck}}$$

$$\frac{}{(\mathsf{read}\ c)\ \mathsf{stuck}} \qquad \frac{R\ \mathsf{stuck}}{(x : \tau \leftarrow R;\ S)\ \mathsf{stuck}}$$

$$\boxed{R \text{ final}}$$

$$\frac{}{(\mathsf{val}\ v)\ \mathsf{final}} \qquad \frac{R\ \mathsf{stuck}}{R\ \mathsf{final}}$$

$$\boxed{R \Downarrow \eta}$$

$$\frac{R \to \sum_i c_i\, 1[R_i] \qquad R_i \Downarrow \eta_i}{R \Downarrow \sum_i c_i\, \eta_i}$$

$$\frac{R\ \mathsf{final}}{R \Downarrow 1[R]}$$

Figure 16: Big-step operational semantics for IPDL reactions.

$$\boxed{P \text{ final}}$$

$$\frac{}{\mathsf{0}\ \mathsf{final}} \qquad \frac{}{(o := v)\ \mathsf{final}} \qquad \frac{R\ \mathsf{stuck}}{(o := R)\ \mathsf{final}} \qquad \frac{P\ \mathsf{final} \qquad Q\ \mathsf{final}}{(P \mid\mid Q)\ \mathsf{final}} \qquad \frac{P\ \mathsf{final}}{(\mathsf{new}\ o : \tau\ \mathsf{in}\ P)\ \mathsf{final}}$$

$$\boxed{P \Downarrow \eta}$$

$$\frac{P \to \sum_i c_i\ 1[P_i] \qquad P_i \Downarrow \eta_i}{P \Downarrow \sum_i c_i\ \eta_i}$$

$$\frac{P \xrightarrow{o := v} Q \qquad Q \Downarrow \eta}{P \Downarrow \eta} \qquad \frac{P\ \mathsf{final}}{P \Downarrow 1[P]}$$

Figure 17: Big-step operational semantics for IPDL protocols.

$$\frac{\mathsf{d}_1 : \sigma_1 \nleftarrow \tau_1, \mathsf{d}_2 : \sigma_2 \nleftarrow \tau_2 \in \Sigma \qquad \Gamma \vdash e_1 : \sigma_1 \qquad \Gamma \vdash e_2 : \sigma_2}{\begin{array}{l}\Delta;\ \Gamma \vdash \big(x_1 : \tau_1 \leftarrow \mathsf{samp}\ (\mathsf{d}_1\ e_1);\ x_2 : \tau_2 \leftarrow \mathsf{samp}\ (\mathsf{d}_2\ e_2);\ \mathsf{ret}\ (x_1, x_2)\big) = \\ \big(x_2 : \tau_2 \leftarrow \mathsf{samp}\ (\mathsf{d}_2\ e_2);\ x_1 : \tau_1 \leftarrow \mathsf{samp}\ (\mathsf{d}_1\ e_1);\ \mathsf{ret}\ (x_1, x_2)\big) : I \to \tau_1 \times \tau_2\end{array}}\ \textsc{exch-samp-samp}$$

$$\frac{\mathsf{d} : \sigma \to \tau_1 \in \Sigma \qquad \Gamma \vdash e : \sigma \qquad i : \tau_2 \in \Delta \qquad i \in I}{\begin{array}{l}\Delta;\ \Gamma \vdash \big(x_1 : \tau_1 \nleftarrow \mathsf{samp}\ (\mathsf{d}\ e);\ x_2 : \tau_2 \leftarrow \mathsf{read}\ i;\ \mathsf{ret}\ (x_1, x_2)\big) = \\ \big(x_2 : \tau_2 \leftarrow \mathsf{read}\ i;\ x_1 : \tau_1 \leftarrow \mathsf{samp}\ (\mathsf{d}\ e);\ \mathsf{ret}\ (x_1, x_2)\big) : I \to \tau_1 \times \tau_2\end{array}}\ \textsc{exch-samp-read}$$

$$\frac{i_1 : \tau_1, i_2 : \tau_2 \in \Delta \qquad i_1, i_2 \in I}{\begin{array}{l}\Delta;\ \Gamma \vdash \big(x_1 : \tau_1 \leftarrow \mathsf{read}\ i_1;\ x_2 : \tau_2 \leftarrow \mathsf{read}\ i_2;\ \mathsf{ret}\ (x_1, x_2)\big) = \\ \big(x_2 : \tau_2 \leftarrow \mathsf{read}\ i_2;\ x_1 : \tau_1 \leftarrow \mathsf{read}\ i_1;\ \mathsf{ret}\ (x_1, x_2)\big) : I \to \tau_1 \times \tau_2\end{array}}\ \textsc{exch-read-read}$$

Figure 18: Alternative formulation of the EXCH rule for reaction equality.

$$\frac{o \notin I \qquad b \in I \qquad b : \mathsf{Bool}, o : \tau \in \Delta \qquad \Delta;\ \cdot \vdash S_1 : I \cup \{o\} \to \tau \qquad \Delta;\ \cdot \vdash S_2 : I \cup \{o\} \to \tau}{\begin{array}{l}\Delta \vdash \big(\mathsf{new}\ l : \tau\ \mathsf{in}\ o := x : \mathsf{Bool} \leftarrow \mathsf{read}\ b;\ \mathsf{if}\ x\ \mathsf{then}\ \mathsf{read}\ l\ \mathsf{else}\ S_2 \parallel l := x : \mathsf{Bool} \leftarrow \mathsf{read}\ b;\ S_1\big) = \\ \big(o := x : \mathsf{Bool} \leftarrow \mathsf{read}\ b;\ \mathsf{if}\ x\ \mathsf{then}\ S_1\ \mathsf{else}\ S_2\big) : I \to \{o\}\end{array}}\ \textsc{fold-if-left}$$

$$\frac{o \notin I \qquad b \in I \qquad b : \mathsf{Bool}, o : \tau \in \Delta \qquad \Delta;\ \cdot \vdash S_1 : I \cup \{o\} \to \tau \qquad \Delta;\ \cdot \vdash S_2 : I \cup \{o\} \to \tau}{\begin{array}{l}\Delta \vdash \big(\mathsf{new}\ r : \tau\ \mathsf{in}\ o := x : \mathsf{Bool} \leftarrow \mathsf{read}\ b;\ \mathsf{if}\ x\ \mathsf{then}\ S_1\ \mathsf{else}\ \mathsf{read}\ r \parallel r := x : \mathsf{Bool} \leftarrow \mathsf{read}\ b;\ S_2\big) = \\ \big(o := x : \mathsf{Bool} \leftarrow \mathsf{read}\ b;\ \mathsf{if}\ x\ \mathsf{then}\ S_1\ \mathsf{else}\ S_2\big) : I \to \{o\}\end{array}}\ \textsc{fold-if-right}$$

Figure 19: Alternative formulation of the FOLD-IF-LEFT and FOLD-IF-RIGHT rules.