

A Logic for Constrained Process Calculi with Applications to Molecular Biology

Kaustuv Chaudhuri
INRIA
kaustuv.chaudhuri@inria.fr

Joëlle Despeyroux
INRIA
joelle.despeyroux@inria.fr

Rapport de recherche INRIA-HAL nb XXX — draft of May 25, 2009 — 28 pages

Abstract

Linear implication can represent state transitions, but real transition systems operate under temporal, stochastic or probabilistic constraints that are not directly representable in ordinary linear logic. We propose a general modal extension of intuitionistic linear logic where logical truth is indexed by constraints and hybrid connectives combine constraint reasoning with logical reasoning. The logic has a focused cut-free sequent calculus that can be used to internalize the rules of particular constrained transition systems; we illustrate this with an adequate encoding of the synchronous stochastic pi-calculus that has been used to encode a number of biochemical reaction systems. We also discuss direct encodings of biochemical reactions in suitable (temporal and stochastic) fragments of the logic.

1 Introduction

To reason about state transition systems, we need a logic of state. Linear logic [15] is such a logic and has been successfully used to model such diverse systems as: planning [36], Petri nets, CCS, the π -calculus [5, 23], concurrent ML [5], security protocols [3], multi-set rewriting, graph traversal algorithms [37], and games. Linear logic achieves this versatility by representing propositions as *resources* that are composed into elements of state using \otimes , which can then be transformed using the linear implication (\multimap). However, linear implication is timeless: there is no way to correlate the result of two concurrent transitions. If resources have lifetimes and state changes have temporal, probabilistic or stochastic *constraints*, then the logic allows inferences that may not be realizable in the system. Molecular biology has a wealth of examples of such systems. In a biochemical reaction, molecules can interact to form other molecules or undergo internal changes such as phosphorylation, and these changes usually occur as parts of networks of interacting processes with continuous kinetic feedback. The need for formal reasoning in such systems has led to the creation of specialized logics such as Continuous Stochastic Logic (CSL) [2] or Probabilistic CTL [17] that pay a considerable encoding overhead for the state component of the transition system in exchange for the constraint reasoning not provided by linear logic.

A prominent alternative to the logical approach is to use a suitably enriched process algebra; a short list of examples includes reversible CCS [10], bioambients [34], brane calculi [4], stochastic and probabilistic π -calculi, the PEPA algebra [18], and the κ -calculus [11]. Each process algebra comes equipped with an underlying algebraic semantics which is used to justify mechanistic abstractions of observed reality as processes. These abstractions are then animated by means of simulation and then compared with the observations. Process calculi do not however fill the need for formal logical reasoning in and about constrained transition systems; for example, there is no uniform language to encode and compare different stochastic process algebras. (Encoding the stochastic π calculus in CSL, for example, would be inordinately complex because CSL does not provide any direct means of encoding π -calculus dynamics such as the linear production and consumption of messages.)

In this paper, we propose a simple yet general method to add constraint reasoning to linear logic, reuniting linguistic need with ability. We extend ordinary logical truth by parameterizing it on a *constraint domain*: $A@w$ stands for the proposition A being true with constraint w . Connectives from hybrid logic are used to perform a limited form

of symbolic reasoning on the constraints at the propositional level. We call the result *hybrid linear logic* (HyLL). No assumptions—except a basic monoidal structure—are made about the constraints. The HyLL proof theory is well behaved: it has a generic cut-free (but cut admitting) sequent calculus that can be strengthened with a focusing restriction [1] to obtain the structural guarantees needed for *representationally adequate* encodings of transition systems. Sequent calculi with cut admissibility and focusing should, we believe, be seen as basic requirements for any linguistic formalisms called *logics*, yet modal logics generally lack them.

HyLL is proposed as a *logical framework* for constrained transition systems. We illustrate this proposal by giving an adequate encoding of the synchronous stochastic π -calculus ($S\pi$) in HyLL where the constraint domains are rate functions. In addition to the stochastic component, our encoding is a conceptual improvement over other existing encodings of process calculi such as the π calculus in linear logic [5, 23]: our encoding performs a full propositional reflection of processes as in [23], but is first-order and adequate as in [5]. Being a logical framework, HyLL does not itself prescribe an operational semantics for the encoding of processes; thus, bisimilarity in CTMCs is not the same as logical equivalence in stochastic HyLL, unlike in CSL [12]. This is not a deficiency; the *combination* of focused HyLL proofs and a proof search strategy tailored to a particular encoding is necessary to produce faithful symbolic executions. This is exactly analogous to $S\pi$ where it is the simulation rather than the transitions in the process calculus that is shown to be faithful to the CTMC semantics [28].

This proposed treatment of *proof search as simulation* is novel. Proof search is traditionally concerned with logical completeness and deals with the inherent non-determinism in the application of inference rules. However, in constrained process calculi not every logically allowed execution is correct, so one must replace the general algorithm with a specific one that produces only correct executions. The points where such a specific algorithm “plugs in” to a more general search algorithm are easily defined (see sec. 4.2). The product of the combined approach is still a proof that represents an execution trace, and can be checked by a traditional proof checker – one need not trust the simulator to be sound.

The proposed logic is a first step towards bringing systems biology into the descriptive sphere of linear logic. Note that, unlike formalisms such as the brane or κ -calculi, we do not propose a new idealised view of biology; instead, as far as system biology is concerned, our proposal should be seen as a uniform language to encode biological systems; providing genuine means to reason about them is left for future work.

The sections of this paper are organized as follows: in sec. 2, we present the syntax and rules (natural deduction and sequent calculus) for HyLL, and give two examples of instances of HyLL for temporal and stochastic constraints. A simple example from systems biology (a “repressilator” network) is given in each instance of HyLL. In sec. 3 we sketch the general focusing restriction on HyLL sequent proofs. In sec. 4 we give the encoding of $S\pi$ in stochastic HyLL, and give a transparently correct adequacy argument (theorems 22 and 24). We end with a discussion of related and future work (sec. 5).

2 Hybrid linear logic

In this section we define HyLL, a conservative extension of intuitionistic first-order linear logic (ILL) [15] where the truth judgements are labelled by *worlds*. Like in ILL, propositions are interpreted as *resources* which may be composed into a *state* using the usual linear connectives, and the linear implication \multimap denotes a transition between states. The world label of a judgement represents a constraint on states and state transitions; particular choices for the worlds produce particular instances of HyLL. The common component in all the instances of HyLL is the proof theory, which we fix once and for all. We impose the following minimal requirements on the constraint domains.

Definition 1. A constraint domain \mathcal{W} is a non-commutative monoid structure $\langle W, \cdot, \iota \rangle$. The elements of W are called worlds, and the partial order $\leq : W \times W$ —defined as $u \leq w$ if there exists $v \in W$ such that $u \cdot v = w$ —is the reachability relation in \mathcal{W} .

The identity world ι is \leq -initial and is intended to represent the lack of any constraints. Thus, the ordinary ILL is embeddable into any instance of HyLL. When needed to disambiguate, the instance of HyLL for the constraint domain \mathcal{W} will be written $\text{HyLL}(\mathcal{W})$.

Atomic propositions are written using minuscule letters (a, b, \dots) applied to a sequence of *terms* (s, t, \dots), which are drawn from an untyped term language containing term variables (x, y, \dots) and function symbols (f, g, \dots) applied to a list of terms.. Non-atomic propositions are constructed from the connectives of first-order intuitionistic linear logic

and the two hybrid connectives *satisfaction* (at), which states that a proposition is true at a given world (w, u, v, \dots) , and *localization* (\downarrow), which binds a name for the world of the proposition. The following grammar summarizes the syntax of HyLL propositions.

$$A, B, \dots ::= a \vec{r} \mid A \otimes B \mid \mathbf{1} \mid A \multimap B \mid A \& B \mid \top \mid A \oplus B \mid \mathbf{0} \mid !A \mid \forall x. A \mid \exists x. A \\ \mid (A \text{ at } w) \mid \downarrow u. A \mid \forall u. A \mid \exists u. A$$

Note that in the propositions $\downarrow u. A$, $\forall u. A$ and $\exists u. A$, the scope of the world variable u is all the worlds occurring in A . World variables cannot be used in terms, and neither can term variables occur in worlds. We let α range over variables of either kind.

The unrestricted connectives \wedge, \vee, \supset , etc. of intuitionistic logic can also be defined in terms of the linear connectives and the exponential $!$ using any of the available embeddings of intuitionistic logic into linear logic, such as Girard's embedding [15].

2.1 Natural deduction for HyLL

We start with the judgements from linear logic [16] and enrich them with a modal situated truth. We present the syntax of hybrid linear logic in a natural deduction style, using Martin-Löf's principle of separating judgements and logical connectives. Instead of the ordinary mathematical judgement "A is true", judgements of HyLL are of the form "A is true at world w ", abbreviated as $A@w$. We use dyadic hypothetical derivations of the form $\Gamma; \Delta \vdash C@w$ where Γ and Δ are sets of judgements of the form $A@w$, with Δ being moreover a *multiset*. Γ is called the *unrestricted context*: its hypotheses can be consumed any number of times. Δ is a *linear context*: every hypothesis in it must be consumed singly in the proof.

The rules for the linear connectives are borrowed from [7] where they are discussed at length, so we omit a more thorough discussion here. The rules for the first-order quantifiers are completely standard. The unrestricted context Γ enjoys weakening and contraction; as usual, this is a theorem that is attested by the inference rules of the logic, and we omit its straightforward inductive proof. The notation $[w/u]A$ stands for the replacement of all free occurrences of the world variable u in A with the world w , avoiding capture.

Theorem 2 (structural properties).

1. If $\Gamma; \Delta \vdash C@w$, then $\Gamma, \Gamma'; \Delta \vdash C@w$. (weakening)
2. If $\Gamma, A@u, A@u; \Delta \vdash C@w$, then $\Gamma, A@u; \Delta \vdash C@w$. (contraction)

The full collection of inference rules are in fig. 1. A brief discussion of the hybrid rules follows. To introduce the *satisfaction* proposition ($A \text{ at } w$) (at any world w'), the proposition A must be true in the world w . The proposition ($A \text{ at } w$) itself is then true at any world, not just in the world w . In other words, ($A \text{ at } w$) carries with it the world at which it is true. Therefore, suppose we know that ($A \text{ at } w$) is true (at any world w'); then, we also know that $A@w$. These two introduction and elimination rules match up precisely to (de)construct the information in the $A@w$ judgement. The other hybrid connective of *localisation*, \downarrow , is intended to be able to name the current world. That is, if $\downarrow u. A$ is true at world w , then the variable u stands for w in the body A . This interpretation is reflected in its introduction rule $\downarrow I$. For elimination, suppose we have a proof of $\downarrow u. A@w$ for some world w . Then, we also know $[w/u]A@w$.

For the linear and unrestricted hypotheses, substitution is no different from that of the usual linear logic.

Theorem 3 (substitution).

1. If $\Gamma; \Delta \vdash A@u$ and $\Gamma; \Delta', A@u \vdash C@w$, then $\Gamma; \Delta, \Delta' \vdash C@w$.
2. If $\Gamma; \cdot \vdash A@u$ and $\Gamma, A@u; \Delta \vdash C@w$, then $\Gamma; \Delta \vdash C@w$.

Proof sketch. By structural induction on the second given derivation in each case. □

Note that the \downarrow connective commutes with every propositional connective, including itself. That is, $\downarrow u. (A * B)$ is equivalent to $(\downarrow u. A) * (\downarrow u. B)$ for all binary connectives $*$, and $\downarrow u. *A$ is equivalent to $*(\downarrow u. A)$ for every unary connective $*$, assuming the commutation will not cause an unsound capture of u . It is purely a matter of taste where to place the \downarrow , and repetitions are harmless.

Theorem 4 (conservativity). *Call a proposition or multiset of propositions pure if it contains no instance of the hybrid connectives, and let Γ, Δ and A be pure. Then, $\Gamma; \Delta \vdash A@w$ in HyLL iff $\Gamma; \Delta \vdash A$ in intuitionistic linear logic.*

Judgemental rules

$$\frac{}{\Gamma; A@w \vdash A@w} \text{hyp} \quad \frac{}{\Gamma, A@w; \cdot \vdash A@w} \text{hyp!}$$

Multiplicative rules

$$\frac{\Gamma; \Delta \vdash A@w \quad \Gamma; \Delta' \vdash B@w}{\Gamma; \Delta, \Delta' \vdash A \otimes B@w} \otimes I \quad \frac{\Gamma; \Delta \vdash A \otimes B@w}{\Gamma; \Delta', A@w, B@w \vdash C@w'} \otimes E$$

$$\frac{}{\Gamma; \cdot \vdash \mathbf{1}@w} \mathbf{1}I \quad \frac{\Gamma; \Delta \vdash \mathbf{1}@w \quad \Gamma; \Delta' \vdash C@w'}{\Gamma; \Delta, \Delta' \vdash C@w'} \mathbf{1}E$$

$$\frac{\Gamma; \Delta, A@w \vdash B@w}{\Gamma; \Delta \vdash A \multimap B@w} \multimap I \quad \frac{\Gamma; \Delta \vdash A \multimap B@w \quad \Gamma; \Delta' \vdash A@w}{\Gamma; \Delta, \Delta' \vdash B@w} \multimap E$$

Additive rules

$$\frac{\Gamma; \Delta \vdash A@w \quad \Gamma; \Delta \vdash B@w}{\Gamma; \Delta \vdash A \& B@w} \&I \quad \frac{\Gamma; \Delta \vdash A_1 \& A_2@w}{\Gamma; \Delta \vdash A_i@w} \&E_i$$

$$\frac{\Gamma; \Delta \vdash A_i@w}{\Gamma; \Delta \vdash A_1 \oplus A_2@w} \oplus I_i \quad \frac{\Gamma; \Delta', A@w \vdash C@w' \quad \Gamma; \Delta', B@w \vdash C@w'}{\Gamma; \Delta, \Delta' \vdash C@w'} \oplus E$$

$$\frac{}{\Gamma; \Delta \vdash \top@w} \top I \quad \frac{\Gamma; \Delta \vdash \mathbf{0}@w}{\Gamma; \Delta, \Delta' \vdash C@w'} \mathbf{0}E$$

First-order rules

$$\frac{\Gamma; \Delta \vdash A@w}{\Gamma; \Delta \vdash \forall \alpha. A@w} \forall I^\alpha \quad \frac{\Gamma; \Delta \vdash \forall \alpha. A@w}{\Gamma; \Delta \vdash [\tau/x]A@w} \forall E$$

$$\frac{\Gamma; \Delta \vdash [\tau/x]A@w}{\Gamma; \Delta \vdash \exists \alpha. A@w} \exists I \quad \frac{\Gamma; \Delta \vdash \exists \alpha. A@w \quad \Gamma; \Delta', A@w \vdash C@w'}{\Gamma; \Delta, \Delta' \vdash C@w'} \exists E^\alpha$$

For $\forall I^\alpha$ and $\exists E^\alpha$, α is assumed to be fresh with respect to the conclusion.
For $\exists I$ and $\forall E$, τ stands for a term or world, as appropriate.

Exponential rules

$$\frac{\Gamma; \cdot \vdash A@w}{\Gamma; \cdot \vdash !A@w} !I \quad \frac{\Gamma; \Delta \vdash !A@w \quad \Gamma, A@w; \Delta' \vdash C@w'}{\Gamma; \Delta, \Delta' \vdash C@w'} !E$$

Hybrid rules

$$\frac{\Gamma; \Delta \vdash A@w}{\Gamma; \Delta \vdash (A \text{ at } w)@w'} \text{at}I \quad \frac{\Gamma; \Delta \vdash (A \text{ at } w)@w'}{\Gamma; \Delta \vdash A@w} \text{at}E$$

$$\frac{\Gamma; \Delta \vdash [w/u]A@w}{\Gamma; \Delta \vdash \downarrow u. A@w} \downarrow J \quad \frac{\Gamma; \Delta \vdash \downarrow u. A@w}{\Gamma; \Delta \vdash [w/u]A@w} \downarrow E$$

Figure 1: Natural deduction for HyLL

Proof. By structural induction on the given HyLL derivation. \square

2.2 Sequent calculus for HyLL

In this section, we give a sequent calculus presentation of HyLL and prove a cut-admissibility theorem. The sequent formulation in turn will lead to an analysis of the polarities of the connectives in order to get a focused sequent calculus that can be used to compile a logical theory into a system of derived inference rules with nice properties (sec. 3). For instance, if a given theory defines a transition system, then the derived rules of the focused calculus will exactly exhibit the same transitions. This is key to obtain the necessary representational adequacy theorems, as we shall see for the $S\pi$ -calculus example chosen in this paper (sec. 4.1).

In the sequent calculus, we depart from the linear hypothetical judgement \vdash which has only an “active” right-hand side to a sequent arrow \Longrightarrow that has active zones on both sides. A rule that infers a proposition on the right of the sequent arrow is called a “right” rule, and corresponds exactly to the introduction rules of natural deduction. Dually, introductions on the left of the sequent arrow correspond to elimination rules of natural deduction; however, as all rules in the sequent calculus are introduction rules, the information flow in a sequent derivation is always in the same direction: from the conclusion to the premises, incidentally making the sequent calculus ideally suited for proof-search.

The full collection of rules of the HyLL sequent calculus is in fig. 2. There are only two structural rules: the init rule infers an atomic initial sequent, and the copy rule introduces a contracted copy of an unrestricted assumption into the linear context (reading from conclusion to premise). Weakening and contraction are admissible rules:

Theorem 5 (structural properties).

1. If $\Gamma ; \Delta \Longrightarrow C@w$, then $\Gamma, \Gamma' ; \Delta \Longrightarrow C@w$. (weakening)
2. If $\Gamma, A@u, A@u ; \Delta \Longrightarrow C@w$, then $\Gamma, A@u ; \Delta \Longrightarrow C@w$. (contraction)

Proof sketch. By straightforward structural induction on the given derivations. \square

The most important structural properties are the admissibility of the identity and the cut principles.

Theorem 6 (identity). $\Gamma ; A@w \Longrightarrow A@w$.

Proof. By induction on the structure of A (see sec. A.1). \square

The identity theorem is the general case of the init rule and serves as a global syntactic completeness theorem for the logic. Dually, the cut theorem below establishes the syntactic soundness of the calculus; moreover there is no cut-free derivation of $\cdot ; \cdot \Longrightarrow \mathbf{0}@w$, so the logic is also globally consistent.

Theorem 7 (cut).

1. If $\Gamma ; \Delta \Longrightarrow A@u$ and $\Gamma ; \Delta', A@u \Longrightarrow C@w$, then $\Gamma ; \Delta, \Delta' \Longrightarrow C@w$.
2. If $\Gamma ; \cdot \Longrightarrow A@u$ and $\Gamma, A@u ; \Delta \Longrightarrow C@w$, then $\Gamma ; \Delta \Longrightarrow C@w$.

Proof. Lexicographic structural induction on the given derivations, with cuts of kind 2 additionally allowed to justify cuts of kind 1. The style of proof sometimes goes by the name of *structural cut-elimination* [7]. See sec. A.2 for the details. \square

We can use the admissible cut rules to show that the following rules are invertible: $\otimes L$, $\mathbf{1}L$, $\oplus L$, $\mathbf{0}L$, $\exists L$, $\neg R$, $\&R$, $\top R$, and $\forall R$. In addition, the four hybrid rules, $\text{at}R$, $\text{at}L$, $\downarrow R$ and $\downarrow L$ are invertible. In fact, \downarrow and at commute freely with all non-hybrid connectives:

Theorem 8 (Invertibility). *The following rules are invertible:*

1. On the right: $\&R$, $\top R$, $\neg R$, $\forall R$, $\downarrow R$ and $\text{at}R$;
2. On the left: $\otimes L$, $\mathbf{1}L$, $\oplus L$, $\mathbf{0}L$, $\exists L$, $\downarrow L$ and $\text{at}L$.

Proof. See §A.3. \square

Theorem 9 (Correctness of the sequent calculus).

Judgemental rules

$$\frac{}{\Gamma; a\vec{t}@u \Rightarrow a\vec{t}@u} \text{init} \quad \frac{\Gamma, A@u; \Delta, A@u \Rightarrow C@w}{\Gamma, A@u; \Delta \Rightarrow C@w} \text{copy}$$

Multiplicatives

$$\frac{\Gamma; \Delta \Rightarrow A@w \quad \Gamma; \Delta' \Rightarrow B@w}{\Gamma; \Delta, \Delta' \Rightarrow A \otimes B@w} \otimes R \quad \frac{\Gamma; \Delta, A@u, B@u \Rightarrow C@w}{\Gamma; \Delta, A \otimes B@u \Rightarrow C@w} \otimes L$$

$$\frac{}{\Gamma; \cdot \Rightarrow \mathbf{1}@w} \mathbf{1}R \quad \frac{\Gamma; \Delta \Rightarrow C@w}{\Gamma; \Delta, \mathbf{1}@u \Rightarrow C@w} \mathbf{1}L \quad \frac{\Gamma; \Delta, A@w \Rightarrow B@w}{\Gamma; \Delta \Rightarrow A \multimap B@w} \multimap R$$

$$\frac{\Gamma; \Delta \Rightarrow A@u \quad \Gamma; \Delta', B@u \Rightarrow C@w}{\Gamma; \Delta, \Delta', A \multimap B@u \Rightarrow C@w} \multimap L$$

Additives

$$\frac{}{\Gamma; \Delta \Rightarrow \top@w} \top R \quad \frac{\Gamma; \Delta \Rightarrow A@w \quad \Gamma; \Delta \Rightarrow B@w}{\Gamma; \Delta \Rightarrow A \& B@w} \& R \quad \frac{\Gamma; \Delta, A_i@u \Rightarrow C@w}{\Gamma; \Delta, \Delta', A_1 \& A_2@u \Rightarrow C@w} \& L_i$$

$$\frac{\Gamma; \Delta \Rightarrow A_i@w}{\Gamma; \Delta \Rightarrow A_1 \oplus A_2@w} \oplus R_i \quad \frac{}{\Gamma; \Delta, \mathbf{0}@u \Rightarrow C@w} \mathbf{0}L \quad \frac{\Gamma; \Delta, A@u \Rightarrow C@w \quad \Gamma; \Delta, B@u \Rightarrow C@w}{\Gamma; \Delta, A \oplus B@u \Rightarrow C@w} \oplus L$$

Quantifiers

$$\frac{\Gamma; \Delta \Rightarrow A@w}{\Gamma; \Delta \Rightarrow \forall \alpha. A@w} \forall R^\alpha \quad \frac{\Gamma; \Delta, [\tau/\alpha]A@u \Rightarrow C@w}{\Gamma; \Delta, \forall \alpha. A@u \Rightarrow C@w} \forall L$$

$$\frac{\Gamma; \Delta \Rightarrow [\tau/\alpha]A@w}{\Gamma; \Delta \Rightarrow \exists \alpha. A@w} \exists R \quad \frac{\Gamma; \Delta, A@u \Rightarrow C@w}{\Gamma; \Delta, \exists \alpha. A@u \Rightarrow C@w} \exists L^\alpha$$

For $\forall R^\alpha$ and $\exists L^\alpha$, α is assumed to be fresh with respect to the conclusion. For $\exists R$ and $\forall L$, τ stands for a term or world, as appropriate.

Exponentials

$$\frac{\Gamma; \cdot \Rightarrow A@w}{\Gamma; \cdot \Rightarrow !A@w} !R \quad \frac{\Gamma, A@u; \Delta \Rightarrow C@w}{\Gamma; \Delta, !A@u \Rightarrow C@w} !L$$

Hybrid connectives

$$\frac{\Gamma; \Delta \Rightarrow A@u}{\Gamma; \Delta \Rightarrow (A \text{ at } u)@v} \text{at}R \quad \frac{\Gamma; \Delta, A@u \Rightarrow C@w}{\Gamma; \Delta, (A \text{ at } u)@v \Rightarrow C@w} \text{at}L$$

$$\frac{\Gamma; \Delta \Rightarrow [w/u]A@w}{\Gamma; \Delta \Rightarrow \downarrow u. A@w} \downarrow R \quad \frac{\Gamma; \Delta, [v/u]A@v \Rightarrow C@w}{\Gamma; \Delta, \downarrow u. A@v \Rightarrow C@w} \downarrow L$$

Figure 2: The sequent calculus for HyLL

1. If $\Gamma ; \Delta \Longrightarrow C@w$, then $\Gamma ; \Delta \vdash C@w$. (soundness)
2. If $\Gamma ; \Delta \vdash C@w$, then $\Gamma ; \Delta \Longrightarrow C@w$. (completeness)

Proof. See §A.4. □

Corollary 10 (consistency). *There is no proof of $\cdot ; \cdot \vdash \mathbf{0}@w$.*

Proof. See §A.4. □

HyLL is conservative with respect to ordinary intuitionistic logic: as long as no hybrid connectives are used, the proofs in HyLL are identical to those in ILL [7]. The proof (omitted) is by simple structural induction.

Theorem 11 (conservativity). *If $\Gamma ; \Delta \Longrightarrow_{\text{HyLL}} C@w$ is derivable, contains no occurrence of the hybrid connectives \downarrow , at , $\forall u$ or $\exists u$, and each element of Γ and Δ is of the form $A@w$, then $\Gamma ; \Delta \Longrightarrow_{\text{ILL}} C$.*

Theorem 12 (HyLL is S5). *The following sequent is derivable: $\cdot ; \diamond A@w \Longrightarrow \square \diamond A@w$.*

Proof. See §A.5. □

In the rest of this paper we make use of the following definable connectives.

Definition 13 (modal connectives).

$$\begin{aligned} \square A &\triangleq \downarrow u. \forall w. (A \text{ at } u \cdot w) & \diamond A &\triangleq \downarrow u. \exists w. (A \text{ at } u \cdot w) \\ \rho_v A &\triangleq \downarrow u. (A \text{ at } u \cdot v) & \dagger A &\triangleq \forall u. (A \text{ at } u) \end{aligned}$$

The connective ρ represents a form of delay. Note its derived right rule:

$$\frac{\Gamma ; \Delta \vdash A@w \cdot v}{\Gamma ; \Delta \vdash \rho_v A@w} \rho R$$

The proposition $\rho_v A$ thus stands for an *intermediate state* in a transition to A . Informally it can be thought to be “ v before A ”; thus, $\forall v. \rho_v A$ represents *all* intermediate states in the path to A , and $\exists v. \rho_v A$ represents *some* such state. The modally unrestricted proposition $\dagger A$ represents a resource that is consumable in any world; it is used to make transition rules applicable at all constrained worlds.

2.3 Temporal semantics

Consider the constraint domain $\mathcal{T} = \langle \mathbb{R}^+, +, 0 \rangle$ representing instants of time. This domain can be used to define the lifetime of resources, such as keys, sessions, or delegations of authority. Delay in $\text{HyLL}(\mathcal{T})$ represents intervals of time; $\rho_d A$ means “ A will become available after delay d ”, similar to metric tense logic [31]. This domain is very permissive because addition is commutative, resulting in the equivalence of $\rho_u \rho_v A$ and $\rho_v \rho_u A$. The “forward-looking” connectives G and F of ordinary tense logic are precisely \square and \diamond of defn. 13. In addition to the future connectives, this domain also admits past connectives if we add saturating subtraction (*i.e.*, $a - b = 0$ if $b \geq a$) to the language of worlds. We can then define the duals H and P of G and F as:

$$\begin{aligned} HA &\triangleq \downarrow u. \forall w. (A \text{ at } u - w) \\ PA &\triangleq \downarrow u. \exists w. (A \text{ at } u - w) \end{aligned}$$

While this domain does not have any branching structure like CTL, it is expressive enough for many common idioms because of the branching structure of derivations involving \oplus . CTL reachability (“in some path in some future”), for instance, is the same as our \diamond ; similarly, CTL stability (“in all paths in all futures”) is the same as \square . There is some loss of expressive power, however; for instance, in CTL steadiness (“in some path for all futures”) is distinct from stability, whereas the best approximation in HyLL is $\exists w. \square(A \text{ at } u \cdot w)$.

On the other hand, the availability of linear reasoning makes certain kinds of reasoning in HyLL much more natural than in ordinary temporal logics. One important example is of *oscillation* between states in systems with kinetic feedback. In a temporal specification language such as BIOCHAM [6], only finite oscillations are representable using a

nested syntax, while in HyLL we use a simple bi-implication; for example, the oscillation between A and B with delay d is represented by the rule $\dagger(A \multimap \rho_d B) \& (B \multimap \rho_d A)$ (or $\dagger(A \multimap \diamond B) \& (B \multimap \diamond A)$ if the oscillation is aperiodic).

If $\text{HyLL}(\mathcal{T})$ were also to have constrained implication and conjunction in the style of CILL [36] or η [13], then we can define localized versions of \square and \diamond , such as “ A is true everywhere/somewhere in an interval”. They would also allow us to define the “until” and “since” operators of linear temporal logic [20].

Example: the repressor To illustrate the temporal semantics, we consider a simplified *repressor* gene network consisting of two genes, each causing the production of a protein that represses the other gene by negative feedback. This is a simplification of the three-gene network constructed in [14]. We note that each gene can be in an “on” (activated) or an “off” (deactivated) state, represented by the unary predicates `on` and `off`. Molecules of the transcribed proteins are represented with the unary predicate `prot`. The system consists of the following components

- *Repression*: Each protein molecule deactivates the next gene in the cycle after (average) deactivation delay d

$$\text{repress } a \ b \stackrel{\text{def}}{=} \text{prot } a \otimes \text{on } b \multimap \rho_d(\text{off } b \otimes \text{prot } a).$$

- *Reactivation*: When a gene is in the “off” state, it eventually becomes “on” after an average delay of r :

$$\text{react} \stackrel{\text{def}}{=} \forall a. \text{off } a \multimap \rho_r \text{on } a.$$

It is precisely this reactivation that causes the system to oscillate instead of being bistable.

- *Transcription*: When a gene is “on”, it transcribes RNA for its protein taking average delay t , after which it continues to be “on” and a molecule of the protein is formed.

$$\text{trans} \stackrel{\text{def}}{=} \forall a. \text{on } a \multimap \rho_t(\text{on } a \otimes \text{prot } a).$$

- *Dissipation*: If a protein does not react with a gene, then it dissipates after average delay s :

$$\text{diss} \stackrel{\text{def}}{=} \forall a. \text{prot } a \multimap \rho_s \mathbf{1}.$$

The system consists of a repression cycle for genes a and b , and the other processes:

$$\text{system} \stackrel{\text{def}}{=} \text{repress } a \ b, \text{repress } b \ a, \text{react}, \text{trans}, \text{diss}.$$

Examples of valid sequents are (0 is the initial instant of time):

$$\dagger \text{system}@0 ; \underbrace{\rho_{t+u} \text{on } a@0, \text{off } b@t}_{\text{initial state}} \Longrightarrow \underbrace{\rho_{u+t+d} \text{off } a \otimes \top@0}_{\text{final state}}$$

From `off b` we get `on b` \otimes `prot b` after interval $t + u$; then `prot b` together with `on a` forms `prot b` \otimes `off a` after a further delay d .

2.4 Rate semantics

Transition systems in practice rarely have precise delays associated with them, so the temporal semantics fails to capture many essential properties. The delay of a transition is determined up to a probability distribution determined over a number of experiments, *i.e.* as a *rate function*.

Definition 14 (rate functions). *A function $r : \mathbb{R}^+ \rightarrow [0, 1]$ is a rate function if $\int_0^\infty r(x) dx \leq 1$. Given two rate functions r_1 and r_2 , their sequence $r_1 \cdot r_2$ is the function: $t \mapsto \int_0^t r_1(x) r_2(t-x) dx$. Let χ_k (for $k \in \mathbb{R}^+$) be the rate function that is 0 everywhere except at k where it is 1.*

It is fairly easy to see that the sequence of two rate functions is a rate function. The sequencing operator is associative with unit χ_0 . The most common rate functions are exponential distributions $x \mapsto ke^{-kx}$ with *rate constant* k . The sequence of a number of exponential distributions produces a well behaved mixture of exponentials that is a rate function with integral 1. Moreover, sequencing is commutative for mixtures of exponentials because they are memoryless.

Definition 15 (HyLL(\mathcal{R})). Let R be the space of rate functions and \mathcal{R} be the constraint domain $\langle R, \cdot, \chi_0 \rangle$. We give the name stochastic hybrid linear logic to the instance HyLL(\mathcal{R}).

Note that sequencing for the χ_k functions is additive: $\chi_j \cdot \chi_k = \chi_{j+k}$. Thus, the temporal semantics is recoverable in the stochastic semantics by defining a “temporal delay” operator $\delta_d A \stackrel{\text{def}}{=} \rho_{\chi_d} A$.

Example: stochastic repressilator We now revisit the example from sec. 2.3, but this time using rates. That is, d, t, r and s are interpreted as rate functions.

$$\begin{aligned} \text{repress } a \ b &\stackrel{\text{def}}{=} \text{prot } a \otimes \text{on } b \multimap \rho_d(\text{off } b \otimes \text{prot } a) \\ \text{trans} &\stackrel{\text{def}}{=} \forall a. \text{on } a \multimap \rho_t(\text{on } a \otimes \text{prot } a). \\ \text{react} &\stackrel{\text{def}}{=} \forall a. \text{off } a \multimap \rho_r \text{on } a. \\ \text{diss} &\stackrel{\text{def}}{=} \forall a. \text{prot } a \multimap \rho_s \mathbf{1}. \end{aligned}$$

Suppose we want to show that in the two-gene repressilator, the state $\text{on}(a) \otimes \text{off}(b)$ can oscillate to $\text{off}(a) \otimes \text{on}(b)$. The proof looks as below, with one named sub-proofs named P , and most of the worlds and a second sub-proof elided:

$$\begin{array}{c} \frac{\frac{\text{off } b \implies \text{off } b}{\text{on } a, \text{off } b \implies \exists k. \rho_k(\text{off } a \otimes \text{on } b)} \text{react} \quad \frac{\frac{\frac{\text{on } b \implies \text{on } b}{\text{on } a, \rho_r \rho_t(\text{on } b \otimes \text{prot } b) \implies \exists k. \rho_k(\text{off } a \otimes \text{on } b)} \text{trans} \quad \frac{\text{on } a, \rho_r \rho_t \text{prot } b \implies \rho_r \rho_t \rho_d \text{off } a \quad \dots}{\text{on } a, \rho_r \rho_t(\text{on } b \otimes \text{prot } b) \implies \exists k. \rho_k(\text{off } a \otimes \text{on } b)} P}{\text{on } a, \rho_r \rho_t \text{prot } b \implies \rho_r \rho_t \text{prot } b} \otimes I}{\text{on } a, \rho_r \rho_t \text{prot } b \implies \rho_r \rho_t \rho_d \text{off } a} \text{repress } b \ a \\ P = \end{array}$$

Note that in this proof we are using the rules at many different worlds; however, this is allowed as the rules are assumed to be modally unrestricted. Another important note is that in the first premise of P we need to show that $\text{on } a \implies \rho_r \rho_t \text{on } a$. This is only possible if the rate of a self-transition on $\text{on } a$ is $r \cdot t$. Of course, this is not derivable from the rest of the theory (and may not actually be true), so it must be added as a new rule; it is the contract that must be satisfied by the repressilator in order for it to oscillate in the desired fashion.

3 Focusing

The HyLL sequent calculus has too many proofs to give operational interpretations to encodings of transition systems. We restrict the syntax to focused derivations [1], which ignores certain irrelevant rule permutations in a sequent proof. The logical connectives are divided into two classes, *negative* and *positive*, and rule permutations for connectives of like polarity are confined to *phases*. A *focused derivation* is one in which the positive and negative rules are applied in alternate maximal *phases* in the following way: in the *active* phase, all negative rules are applied (in irrelevant order) until no further negative rule can apply; the phase then switches and one positive proposition is selected for *focus*; this focused proposition is decomposed under focus (*i.e.*, the focus persists unto its sub-propositions) until it becomes negative, and the phase switches again.

As noted before, the logical rules of the hybrid connectives at and \downarrow are invertible, so they can be considered to have both polarities. It would be valid to decide a polarity for each occurrence of each hybrid connective independently; however, as they are mainly intended for book-keeping during logical reasoning, we define the polarity of these connectives in the following *parasitic* form: if its immediate subformula is positive (resp. negative) connective, then it is itself positive (resp. negative). These connectives therefore become invisible to focusing. This choice of polarity can be seen as a particular instance of a general scheme that divides the \downarrow and at connectives into two polarized forms each.

To complete the picture, we also assign a polarity for the atomic propositions; this can restrict the shape of focusing phases further [9], and is crucial to our intended use of focusing. The full syntax of positive (P, Q, \dots) and negative (M, N, \dots) propositions is as follows:

$$\begin{aligned} P, Q, \dots &::= p \vec{r} \mid P \otimes Q \mid \mathbf{1} \mid P \oplus Q \mid \mathbf{0} \mid !N \mid \exists \alpha. P \mid \Downarrow u. P \mid (P \text{ at } w) \mid \Downarrow N \\ N, M, \dots &::= n \vec{r} \mid N \& N \mid \top \mid P \multimap N \mid \forall \alpha. N \mid \Downarrow u. N \mid (N \text{ at } w) \mid \Uparrow P \end{aligned}$$

The two syntactic classes refer to each other via the new connectives \Uparrow and \Downarrow . Sequents in the focusing calculus are of the following forms.

$$\left. \begin{array}{l} \Gamma ; \Delta ; \Omega \Longrightarrow \cdot ; P @ w \\ \Gamma ; \Delta ; \Omega \Longrightarrow N @ w ; \cdot \end{array} \right\} \text{active sequents} \quad \left. \begin{array}{l} \Gamma ; \Delta ; [N @ u] \Longrightarrow \cdot ; P @ w \\ \Gamma ; \Delta ; [N @ u] \Longrightarrow P @ w ; \cdot \\ \Gamma ; \Delta \Longrightarrow [P @ w] \end{array} \right\} \text{focused sequents}$$

In each case, Γ and Δ contain only negative propositions (*i.e.*, of the form $N @ u$) and Ω only positive propositions (*i.e.*, of the form $P @ u$). The full collection of inference rules are in fig. 3. The sequent form $\Gamma ; \Delta ; \cdot \Longrightarrow \cdot ; P @ w$ is called a *neutral sequent*; from such a sequent, a left or right focused sequent is produced with the rules lf, cplf or rf. Focused logical rules are applied (non-deterministically) and focus persists unto the subformulas of the focused proposition as long as they are of the same polarity; when the polarity switches, the result is an active sequent, where the propositions in the innermost zones are decomposed in an irrelevant order until once again a neutral sequent results.

Soundness of the focusing calculus with respect to the ordinary sequent calculus is immediate by simple structural induction. In each case, if we forget the additional structure in the focused derivations, then we obtain simply an unfocused proof. We omit the obvious theorem. Completeness, on the other hand, is a harder result. A detailed proof of it would be considerably out of the scope of this paper, but it is by now a well known result with a number of distinct proofs (see summary in [8]). The hybrid connectives pose no problems because they allow all permutations in either phase.

Theorem 16 (completeness of focusing). *Let Γ^- and $C^- @ w$ be negative polarizations of Γ and $C @ w$ (that is, adding \Uparrow and \Downarrow s to make C and each proposition in Γ negative) and Δ^+ be a positive polarization of Δ . If $\Gamma ; \Delta \Longrightarrow C @ w$, then $\cdot ; \cdot ; !\Gamma^-, \Delta^+ \Longrightarrow C^- @ w ; \cdot$.*

4 Encoding the stochastic π -calculus

In this section we shall illustrate the use of HyLL(\mathcal{R}) by encoding the stochastic π -calculus ($S\pi$). HyLL(\mathcal{R}) can therefore be seen as a formal language for expressing $S\pi$ executions (traces). For the rest of this section we shall use r, s, t, \dots instead of u, v, w, \dots to highlight the fact that the worlds are rate functions. $S\pi$ extends the ordinary π -calculus by decorating each channel and internal action with an *inherent* rate of synchronization. It has successfully been used in a number of case studies in formal molecular biology, and there are several tools for discrete-time simulation of $S\pi$ processes (SPiM [28], BioPSI, *etc.*). We first briefly summarize the syntax of $S\pi$, which is a minor variant of a number of similar presentations such as [30]. For hygienic reasons we divide entities into the syntactic categories of *processes* (P, Q, \dots) and *sums* (M, N, \dots), defined as follows:

$$\begin{aligned} (\text{Processes}) \quad P, Q, \dots &::= \nu_r P \mid P \mid Q \mid \mathbf{0} \mid X_n x_1 \cdots x_n \mid M \\ (\text{Sums}) \quad M, N, \dots &::= !x(y). P \mid ?x. P \mid \tau_r. P \mid M + N \\ (\text{Environments}) \quad E &::= E, X_n \triangleq P \mid \cdot \end{aligned}$$

$P \mid Q$ is the parallel composition of P and Q , with unit $\mathbf{0}$. The restriction $\nu_r P$ abstracts over a free channel x in the process $P x$. We write the process using higher-order abstract syntax [27], *i.e.*, P is (syntactically) a function from channels to processes. This style lets us avoid cumbersome binding rules in the interactions because we reuse the well-understood binding structure of the λ -calculus. A similar approach was taken in the earliest encoding of (ordinary) π -calculus in (unfocused) linear logic [23], and is also present in the encoding in CLF [5].

A sum is a non-empty choice (+) over terms with *action prefixes*: the output action $!x(y)$ sends y along channel x , the input action $?x$ reads a value from x (which is applied to its continuation process), and the internal action τ_r has no observable I/O behaviour. Replication of processes happens via guarded recursive definitions [24]; in [35] it is argued that they are more practical for programming than the replication operator $!$. In a definition $X_n \triangleq P$, X_n denotes

Focused logical rules

$$\begin{array}{c}
\frac{}{\Gamma; [n \vec{t}@w] \Rightarrow \Downarrow n \vec{t}@w} \text{ li} \quad \frac{\Gamma; \Delta; P@u \Rightarrow \cdot; Q@w}{\Gamma; \Delta; [\uparrow P@u] \Rightarrow Q@w} \uparrow L \quad \frac{\Gamma; \Delta; [N_i@u] \Rightarrow Q@w}{\Gamma; \Delta; [N_1 \& N_2@u] \Rightarrow Q@w} \&L_i \\
\frac{\Gamma; \Delta \Rightarrow [P@u] \quad \Gamma; \Xi; [N@u] \Rightarrow Q@w}{\Gamma; \Delta, \Xi; [P \neg N@u] \Rightarrow Q@w} \neg L \quad \frac{\Gamma; \Delta; [[\tau/\alpha]N@u] \Rightarrow Q@w}{\Gamma; \Delta; [\forall \alpha. N@u] \Rightarrow Q@w} \forall L \\
\frac{\Gamma; \Delta; [[v/u]N@v] \Rightarrow Q@w}{\Gamma; \Delta; [\downarrow u. N@v] \Rightarrow Q@w} \downarrow LF \quad \frac{\Gamma; \Delta; [N@u] \Rightarrow Q@w}{\Gamma; \Delta; [(N \text{ at } u)@v] \Rightarrow Q@w} \text{ at}LF \\
\frac{}{\Gamma; \uparrow p \vec{t}@w \Rightarrow [p \vec{t}@w]} \text{ ri} \quad \frac{\Gamma; \Delta; \cdot \Rightarrow N@w; \cdot}{\Gamma; \Delta \Rightarrow [\Downarrow N@w]} \downarrow R \quad \frac{\Gamma; \Delta \Rightarrow [P@w] \quad \Gamma; \Xi \Rightarrow [Q@w]}{\Gamma; \Delta, \Xi \Rightarrow [P \otimes Q@w]} \otimes R \\
\frac{}{\Gamma; \cdot \Rightarrow [1@w]} \text{ 1R} \quad \frac{\Gamma; \Delta \Rightarrow [P_i@w]}{\Gamma; \Delta \Rightarrow [P_1 \oplus P_2@w]} \oplus R_i \quad \frac{\Gamma; \cdot; \cdot \Rightarrow N@w; \cdot}{\Gamma; \cdot \Rightarrow [!N]@w} !R \\
\frac{\Gamma; \Delta \Rightarrow [[\tau/\alpha]P@w]}{\Gamma; \Delta \Rightarrow [\exists \alpha. P@w]} \exists R \quad \frac{\Gamma; \Delta \Rightarrow [[w/u]P@w]}{\Gamma; \Delta \Rightarrow [\downarrow u. P@w]} \downarrow RF \quad \frac{\Gamma; \Delta \Rightarrow [P@u]}{\Gamma; \Delta \Rightarrow [(P \text{ at } u)@w]} \text{ at}RF
\end{array}$$

Active logical rules (\mathcal{R} of the form $\cdot; Q@w$ or $N@w; \cdot$, and \mathcal{L} of the form $\Gamma; \Delta; \Omega$)

$$\begin{array}{c}
\frac{\mathcal{L}, P@u, Q@u \Rightarrow \mathcal{R}}{\mathcal{L}, P \otimes Q@u \Rightarrow \mathcal{R}} \otimes L \quad \frac{\mathcal{L} \Rightarrow \mathcal{R}}{\mathcal{L}, 1@u \Rightarrow \mathcal{R}} \text{ 1L} \quad \frac{\mathcal{L}, P@u \Rightarrow \mathcal{R} \quad \mathcal{L}, Q@u \Rightarrow \mathcal{R}}{\mathcal{L}, P \oplus Q@u \Rightarrow \mathcal{R}} \oplus L \quad \frac{}{\mathcal{L}, 0@u \Rightarrow \mathcal{R}} \text{ 0L} \\
\frac{\mathcal{L}, [v/u]P@v \Rightarrow \mathcal{R}}{\mathcal{L}, \downarrow u. P@v \Rightarrow \mathcal{R}} \downarrow LA \quad \frac{\mathcal{L}, P@u \Rightarrow \mathcal{R}}{\mathcal{L}, (P \text{ at } u)@v \Rightarrow \mathcal{R}} \text{ at}LA \quad \frac{\mathcal{L}, P@u \Rightarrow \mathcal{R}}{\mathcal{L}, \exists \alpha. P@u \Rightarrow \mathcal{R}} \exists L^\alpha \\
\frac{\Gamma, N@u; \Delta; \Omega \Rightarrow \mathcal{R}}{\Gamma; \Delta; \Omega, !N@u \Rightarrow \mathcal{R}} !L \quad \frac{\Gamma; \Delta, N@w; \Omega \Rightarrow \mathcal{R}}{\Gamma; \Delta; \Omega, \Downarrow N@w \Rightarrow \mathcal{R}} \Downarrow L \quad \frac{\Gamma; \Delta, \uparrow p \vec{t}; \Omega \Rightarrow \mathcal{R}}{\Gamma; \Delta; \Omega, p \vec{t}@w \Rightarrow \mathcal{R}} \text{ lp} \\
\frac{\mathcal{L} \Rightarrow M@w; \cdot \quad \mathcal{L} \Rightarrow N@w; \cdot}{\mathcal{L} \Rightarrow M \& N@w; \cdot} \&R \quad \frac{}{\mathcal{L} \Rightarrow \top@w; \cdot} \top R \quad \frac{\mathcal{L}, P@w \Rightarrow N@w; \cdot}{\mathcal{L} \Rightarrow P \neg N@w; \cdot} \neg R \\
\frac{\mathcal{L} \Rightarrow [w/u]N@w; \cdot}{\mathcal{L} \Rightarrow \downarrow u. N@w; \cdot} \downarrow RA \quad \frac{\mathcal{L} \Rightarrow N@u}{\mathcal{L} \Rightarrow (N \text{ at } u)@w} \text{ at}RA \quad \frac{\mathcal{L} \Rightarrow N@u; \cdot}{\mathcal{L} \Rightarrow \forall \alpha. N@u; \cdot} \forall L^\alpha \\
\frac{\mathcal{L} \Rightarrow \cdot; P@w}{\mathcal{L} \Rightarrow \uparrow P@w; \cdot} \uparrow R \quad \frac{\mathcal{L} \Rightarrow \cdot; \Downarrow n \vec{t}@w}{\mathcal{L} \Rightarrow n \vec{t}@w; \cdot} \text{ rp}
\end{array}$$

Focusing decisions (\mathcal{L} of the form $\Gamma; \Delta$)

$$\begin{array}{c}
\frac{\Gamma; \Delta; [N@u] \Rightarrow Q@w \quad N \text{ not } \uparrow p \vec{t}}{\Gamma; \Delta, N@u; \cdot \Rightarrow \cdot; Q@w} \text{ lf} \quad \frac{\Gamma, N@u; \Delta; [N@u] \Rightarrow Q@w}{\Gamma, N@u; \Delta; \cdot \Rightarrow \cdot; Q@w} \text{ cplf} \\
\frac{\Gamma; \Delta \Rightarrow [P@w] \quad P \text{ not } \Downarrow n \vec{t}}{\Gamma; \Delta; \cdot \Rightarrow \cdot; P@w} \text{ rf}
\end{array}$$

Figure 3: Focusing rules for HyLL.

<i>Interactions</i>			
$\frac{}{!x(y). P + M \mid ?x. Q + M' \xrightarrow{\text{rate}(x)} P \mid Q(y)} \text{ SYN}$		$\frac{}{\tau_r. P \xrightarrow{r} P} \text{ INT}$	
$\frac{P \xrightarrow{r} P'}{P \mid Q \xrightarrow{r} P' \mid Q} \text{ PAR}$	$\frac{\forall x_s. (P(x) \xrightarrow{r} Q(x))}{\nu_s P \xrightarrow{r} \nu_s Q} \text{ RES}$	$\frac{P \xrightarrow{r} Q \quad P \equiv P' \quad Q \equiv Q'}{P' \xrightarrow{r} Q'} \text{ CONG}$	
<i>Congruence</i>			
$\overline{P \mid 0 \equiv P}$	$\overline{P \mid Q \equiv Q \mid P}$	$\overline{P \mid (Q \mid R) \equiv (P \mid Q) \mid R}$	$\overline{\nu_r 0 \equiv 0}$
$\frac{X_n \triangleq P \in E}{E \vdash X_n x_1 \cdots x_n \equiv P x_1 \cdots x_n}$			
$\overline{\nu_r(\lambda x. \nu_s(\lambda y. P)) \equiv \nu_s(\lambda y. \nu_r(\lambda x. P))}$		$\frac{\forall x_r. (P(x) \equiv Q(x))}{\nu_r P \equiv \nu_r Q}$	
$\overline{\nu_r(\lambda x. P \mid Q(x)) \equiv P \mid \nu_r Q}$			
$\frac{P \equiv P'}{P \mid Q \equiv P' \mid Q}$	$\frac{P \equiv P'}{!x(m). P \equiv !x(m). P'}$	$\frac{\forall n. (P(n) \equiv Q(n))}{?x. P \equiv ?x. Q}$	$\frac{P \equiv P'}{\tau_r. P \equiv \tau_r. P'}$
$\overline{M + N \equiv N + M} \quad \overline{M + (N + K) \equiv (M + N) + K} \quad \overline{M + N \equiv M' + N} \quad \overline{M \equiv N}$			

Figure 4: Interactions and congruence in $S\pi$. The environment E is elided in most rules.

a (higher-order) defined constant of arity n ; given channels x_1, \dots, x_n , the process $X_n x_1 \cdots x_n$ is synonymous with $P x_1 \cdots x_n$. The constant X_n may occur on the right hand side of any definition in E , including in its body P , as long as it is prefixed by an action; this prevents infinite recursion without progress.

Interactions are of the form $E \vdash P \xrightarrow{r} Q$ denoting a transition from the process P to the process Q , in a global environment E , by performing an action at rate r . Each channel x is associated with an inherent rate specific to the channel, and internal actions τ_r have rate r . The restriction $\nu_r P$ defines the rate of the abstracted channel as r .

The full set of interactions and congruences are in fig. 4. We generally omit the global environment E in the rules as it never changes. It is possible to use the congruences to compute a normal form for processes that are a parallel composition of sums and each reaction selects two suitable sums to synchronise on a channel until there are no further reactions possible; this refinement of the operational semantics is used in $S\pi$ simulators such as SPiM [29]. We come back to the simulation aspect of $S\pi$ in §4.2.

We begin by encoding the syntax of $S\pi$ as HyLL(\mathcal{R}) propositions.

Definition 17 (syntax encoding).

1. The encoding of the process P as a positive proposition, written $\llbracket P \rrbracket_p$, is as follows (**sel** is a positive atom and **rt** a negative atom).

$$\begin{aligned} \llbracket P \mid Q \rrbracket_p &= \llbracket P \rrbracket_p \otimes \llbracket Q \rrbracket_p & \llbracket \nu_r P \rrbracket_p &= \exists x. !(rt \ x \ \text{at } r) \otimes \llbracket P(x) \rrbracket_p \\ \llbracket 0 \rrbracket_p &= \mathbf{1} & \llbracket X_n x_1 \cdots x_n \rrbracket_p &= X_n x_1 \cdots x_n \\ \llbracket M \rrbracket_p &= \Downarrow(\text{sel} \ \neg \circ \llbracket M \rrbracket_s) \end{aligned}$$

2. The encoding of the sum M as a negative proposition, written $\llbracket M \rrbracket_s$, is as follows (**out**, **in** and **tau** are positive atoms).

$$\begin{aligned} \llbracket M + N \rrbracket_s &= \llbracket M \rrbracket_s \ \& \ \llbracket N \rrbracket_s & \llbracket !x(m). P \rrbracket_s &= \uparrow(\text{out } x \ m \ \otimes \ \llbracket P \rrbracket_p) \\ \llbracket ?x. P \rrbracket_s &= \forall n. (\text{out } x \ n \ \neg \circ \uparrow(\text{in } x \ \otimes \ \llbracket P(n) \rrbracket_p)) & \llbracket \tau_r. P \rrbracket_s &= \uparrow(\text{tau } r \ \otimes \ \llbracket P \rrbracket_p) \end{aligned}$$

3. The encoding of the definitions E as a context, written $\llbracket E \rrbracket_e$, is as follows.

$$\begin{aligned} \llbracket E, X_n \triangleq P \rrbracket_e &= \llbracket E \rrbracket_e \ \dagger \ \forall x_1, \dots, x_n. X_n x_1 \cdots x_n \ \circ \neg \circ \llbracket P x_1 \cdots x_n \rrbracket_p \\ \llbracket \cdot \rrbracket_e &= \cdot \end{aligned}$$

where $P \circ \neg \circ Q$ is defined as $(P \ \neg \circ \uparrow Q) \ \& \ (Q \ \neg \circ \uparrow P)$.

The encoding of processes is positive, so they will be decomposed in the active phase when they occur on the left of the sequent arrow, leaving a collection of sums. The encoding of restrictions will introduce a fresh unrestricted assumption about the rate of the restricted channel. Each sum encoded as a processes undergoes a polarity switch because \multimap is negative; the antecedent of this implication is a *guard* sel . This pattern of guarded switching of polarities prevents unsound congruences such as $!x(m). !y(n). P \equiv !y(n). !x(m). P$ that do not hold for the synchronous π calculus.¹ This selector also *locks* the sums in the context: the $S\pi$ interaction semantics discard the non-interacting terms of a sum only at the time of interaction, so the environment will contain the requisite number of sel s only when an interaction is in progress. The action prefixes themselves are also synchronous, which causes another polarity switch. Each action releases a token of its respective kind— out , in or tau —into the context. These tokens must be consumed before the next interaction: the out token is consumed by the input action, while the other two will be consumed by the interaction rule itself. For each action, the (encoding of the) continuation process is also released into the context.

The proof of the following congruence lemma is omitted. Because the encoding is (essentially) a $\otimes/\&$ structure, there are no applicable distributive laws that would break the process/sum structure.

Theorem 18 (congruence).

$E \vdash P \equiv Q$ iff both $\llbracket E \rrbracket_e @ \chi_0 ; \cdot ; \llbracket P \rrbracket_p @ \chi_0 \Longrightarrow \cdot ; \llbracket Q \rrbracket_p @ \chi_0$ and $\llbracket E \rrbracket_e @ \chi_0 ; \cdot ; \llbracket Q \rrbracket_p @ \chi_0 \Longrightarrow \cdot ; \llbracket P \rrbracket_p @ \chi_0$.

Now we encode the interactions. Because we lift processes into propositions, we can be parsimonious with our encoding and define it only for the atomic interactions syn and int (below); the par , res and cong rules will be ambiently implemented by the logic. Because there are no concurrent interactions—only one interaction can trigger at a time in a trace—the interaction rules must obey a locking discipline. We represent this lock as the proposition act that is consumed at the start of an interaction and produced again at the end. This lock also carries the net rate of the prefix of the trace so far: that is, an interaction $P \xrightarrow{r} Q$ will update the lock from $\text{act} @ s$ to $\text{act} @ s \cdot r$. The encoding of individual atomic interactions must also remove the in , out and tau tokens introduced in context by the interacting processes.

Definition 19 (interaction).

Let $\text{inter} \triangleq \dagger(\text{act} \multimap \uparrow \text{int} \ \& \ \uparrow \text{syn})$ where act is a positive atom and int and syn are as follows:

$$\begin{aligned} \text{int} &\triangleq (\text{sel} \text{ at } \chi_0) \otimes \Downarrow \psi_r. ((\text{tau } r \text{ at } \chi_0) \multimap \rho_r \uparrow \text{act}) \\ \text{syn} &\triangleq (\text{sel} \otimes \text{sel} \text{ at } \chi_0) \otimes \Downarrow \forall x, r. ((\text{in } x \text{ at } \chi_0) \multimap \Downarrow (\text{rt } x \text{ at } r) \multimap \rho_r \uparrow \text{act}). \end{aligned}$$

The number of interactions that are allowed depend on the number of instances of inter in the context. For all finite traces, inter should be added to the unrestricted context so it may be reused as many times as needed.

4.1 Representational adequacy.

Adequacy consists of two components: completeness and soundness. Completeness is the property that every $S\pi$ execution is obtainable as a HyLL derivation using this encoding, and is the comparatively simpler direction (see thm. 22). Soundness is the reverse property, and is false for unfocused HyLL as such. However, it *does* hold for focused proofs (see thm. 24). In both cases, we reason about the following canonical sequents of HyLL.

Definition 20. The canonical context of P , written $\langle P \rangle$, is given by the following equations.

$$\langle X_n x_1 \cdots x_n \rangle = \uparrow X_n x_1 \cdots x_n \quad \langle P \mid Q \rangle = \langle P \rangle, \langle Q \rangle \quad \langle 0 \rangle = \cdot \quad \langle \nu_r P \rangle = \langle P a \rangle \quad \langle M \rangle = \text{sel} \multimap \llbracket M \rrbracket_s$$

For $\langle \nu_r P \rangle$, the right hand side uses a fresh channel a that is not free the rest of the sequent it occurs in.

As an illustration, take $P \triangleq !x(a). Q \mid ?x. R$. We have:

$$\begin{aligned} \langle P \rangle &= \text{sel} \multimap \uparrow (\text{out } x a \otimes \llbracket Q \rrbracket_p), \\ &\quad \text{sel} \multimap \forall y. (\text{out } x y \multimap \uparrow (\text{in } x \otimes \llbracket R y \rrbracket_p)) \end{aligned}$$

Obviously, the canonical context is what would be emitted to the linear zone at the end of the active phase if $\llbracket P \rrbracket_p$ were to be present in the active zone.

¹Note: $(x \multimap a \otimes (x \multimap b \otimes c)) \multimap (x \multimap b \otimes (x \multimap a \otimes c))$ is not provable in linear logic.

Suppose $\mathcal{L} = \text{rt}x@r, \text{inter}@_{\chi_0}$ and $\mathcal{R} = (\llbracket S \rrbracket_p \text{ at } \chi_0) \otimes \text{act}@t$. (All judgements $@_{\chi_0}$ omitted.)

$$\begin{array}{c}
\frac{\mathcal{L}; \mathcal{Q}, \mathcal{R}a, \uparrow \text{act}@s \cdot r; \cdot \Longrightarrow \cdot; \mathcal{R}}{\mathcal{L}; \mathcal{Q}, \uparrow \text{in } x, \mathcal{R}a, \forall x, r. ((\text{in } x \text{ at } \chi_0) \multimap \Downarrow(\text{rt}x \text{ at } r) \multimap \rho_r \text{ act})@s; \cdot \Longrightarrow \cdot; \mathcal{R}} \quad 5 \\
\frac{\mathcal{L}; \uparrow \text{out } x a, \mathcal{Q}, \text{sel} \multimap \forall y. (\text{out } x y \multimap \uparrow(\text{in } x \otimes \llbracket Ry \rrbracket_p)), \uparrow \text{sel}, \forall x, r. ((\text{in } x \text{ at } \chi_0) \multimap \Downarrow(\text{rt}x \text{ at } r) \multimap \rho_r \text{ act})@s; \cdot \Longrightarrow \cdot; \mathcal{R}}{\mathcal{L}; \text{sel} \multimap \uparrow(\text{out } x a \otimes \llbracket Q \rrbracket_p), \text{sel} \multimap \forall y. (\text{out } x y \multimap \uparrow(\text{in } x \otimes \llbracket Ry \rrbracket_p)), \uparrow \text{sel}, \uparrow \text{sel}, \forall x, r. ((\text{in } x \text{ at } \chi_0) \multimap \Downarrow(\text{rt}x \text{ at } r) \multimap \rho_r \text{ act})@s; \cdot \Longrightarrow \cdot; \mathcal{R}} \quad 4 \\
\frac{\mathcal{L}; \text{sel} \multimap \uparrow(\text{out } x a \otimes \llbracket Q \rrbracket_p), \text{sel} \multimap \forall y. (\text{out } x y \multimap \uparrow(\text{in } x \otimes \llbracket Ry \rrbracket_p)), \uparrow \text{sel}, \uparrow \text{sel}, \forall x, r. ((\text{in } x \text{ at } \chi_0) \multimap \Downarrow(\text{rt}x \text{ at } r) \multimap \rho_r \text{ act})@s; \cdot \Longrightarrow \cdot; \mathcal{R}}{\mathcal{L}; \uparrow \text{act}@s, \text{sel} \multimap \uparrow(\text{out } x a \otimes \llbracket Q \rrbracket_p), \text{sel} \multimap \forall y. (\text{out } x y \multimap \uparrow(\text{in } x \otimes \llbracket Ry \rrbracket_p)); [\text{inter}] \Longrightarrow \mathcal{R}} \quad 3 \\
\frac{\mathcal{L}; \uparrow \text{act}@s, \text{sel} \multimap \uparrow(\text{out } x a \otimes \llbracket Q \rrbracket_p), \text{sel} \multimap \forall y. (\text{out } x y \multimap \uparrow(\text{in } x \otimes \llbracket Ry \rrbracket_p)); \cdot \Longrightarrow \cdot; \mathcal{R}}{\mathcal{L}; \uparrow \text{act}@s, \mathcal{Q} | ?x. \mathcal{R}} \quad 2 \\
\frac{\mathcal{L}; \uparrow \text{act}@s, \mathcal{Q} | ?x. \mathcal{R}}{\mathcal{L}; \uparrow \text{act}@s, \mathcal{Q} | ?x. \mathcal{R}} \quad 1
\end{array}$$

Steps

1: focus on $\text{inter} \in \mathcal{L}$	3: sel for output + full phases	5: cleanup
2: select syn from inter , active rules	4: sel for input + full phases	

Figure 5: Example interaction in the $S\pi$ -encoding.

Definition 21. A neutral sequent is canonical if it has the shape

$$\llbracket E \rrbracket_e, \text{rates}, \text{inter}@_{\chi_0}; \uparrow \text{act}@s, \mathcal{P}_1 | \dots | \mathcal{P}_k @_{\chi_0}; \cdot \Longrightarrow \cdot; (\llbracket Q \rrbracket_p \text{ at } \chi_0) \otimes \text{act}@t$$

where rates contains elements of the form $\text{rt}x@r$ defining the rate of the channel x as r , and all free channels in $\llbracket E \rrbracket_e, \mathcal{P}_1 | \dots | \mathcal{P}_k | \mathcal{Q}$ have a single such entry in rates .

Figure 5 contains an example of a derivation for a canonical sequent involving P . Focusing on any (encoding of a) sum in $\mathcal{P}_k @_{\chi_0}$ will fail because there is no sel in the context, so only inter can be given focus; this will consume the act and releasing two copies of $(\text{sel} \text{ at } \chi_0)$ and the continuation into the context. Focusing on the latter will fail now (because $\text{in } x$ is not yet available), so the only applicable foci are the two sums that can now be “unlocked” using the sel s. The output must be unlocked first because the input action depends on the $\text{out } x a$ produced by it. Then the input action is unlocked with the other sel , consuming the $\text{out } a$ and producing $\text{in } x$. This leaves only the continuation for focus, which consumes $\text{in } x$ and emits $\text{act}@s \cdot r$, where r is the inherent rate of x . This sequent is canonical and contains $\mathcal{Q} | \mathcal{R}a$.

Our encoding therefore represents every $S\pi$ action in terms of “micro” actions in the following rigid order order: one micro action to determine what kind of action (internal or synchronization), one micro action per sum to select the term(s) that will interact, and finally one micro action to establish the contract of the action. The micro actions for selections in a synchronization must be performed in the order of output before input. Thus we see that focusing is crucial to maintain the semantic interpretation of (neutral) sequents. In an unfocused calculus, several of these steps could have partial overlaps, making such a semantic interpretation inordinately complicated. We do not know of any encoding of the π calculus that can provide such interpretations in unfocused sequents without changing the underlying logic. In CLF [5] the logic is extended with explicit monadic staging, and this enables a form of adequacy [5]; however, the encoding is considerably more complex because processes and sums cannot be fully lifted and must instead be specified in terms of a lifting computation. Adequacy is then obtained via a permutative equivalence over the lifting operation.

Theorem 22 (completeness). *If $E \vdash P \xrightarrow{r} Q$, then the following canonical sequent is derivable.*

$$\llbracket E \rrbracket_e, \text{rates}, \text{inter}@_{\chi_0}; \uparrow \text{act}@s, \mathcal{P} @_{\chi_0}; \cdot \Longrightarrow \cdot; (\llbracket Q \rrbracket_p \text{ at } \chi_0) \otimes \text{act}@s \cdot r.$$

Proof sketch. By structural induction of the derivation of $E \vdash P \xrightarrow{r} Q$. Every interaction rule of $S\pi$ is implementable as an admissible inference rule for canonical sequents. For cong , we appeal to thm. 18. \square

Completeness is a testament to the expressivity of the logic – all executions of $S\pi$ are also expressible in HyLL. However, we also require the opposite (soundness) direction: that every canonical sequent encodes a possible $S\pi$ trace. The proof hinges on the following canonicity lemma.

Lemma 23 (canonical derivations). *In a derivation for a canonical sequent, the derived inferences between applications of cplf on inter are of one of the following forms (where conclusions and premises are canonical).*

$$\frac{\frac{\frac{\llbracket E \rrbracket_e, \text{rates}, \text{inter}@x_0; \uparrow \text{act}@s, \zeta P \rrbracket_{@x_0}; \cdot \Longrightarrow \cdot; (\llbracket P \rrbracket_p \text{ at } x_0) \otimes \text{act}@s}}{\llbracket E \rrbracket_e, \text{rates}, \text{inter}@x_0; \uparrow \text{act}@s \cdot r, \zeta Q \rrbracket_{@x_0}; \cdot \Longrightarrow \cdot; (\llbracket R \rrbracket_p \text{ at } x_0) \otimes \text{act}@t}}{\llbracket E \rrbracket_e, \text{rates}, \text{inter}@x_0; \uparrow \text{act}@s, \zeta P \rrbracket_{@x_0}; \cdot \Longrightarrow \cdot; (\llbracket R \rrbracket_p \text{ at } x_0) \otimes \text{act}@t}}}{\llbracket E \rrbracket_e, \text{rates}, \text{inter}@x_0; \uparrow \text{act}@s, \zeta P \rrbracket_{@x_0}; \cdot \Longrightarrow \cdot; (\llbracket R \rrbracket_p \text{ at } x_0) \otimes \text{act}@t}}$$

where: either $E \vdash P \xrightarrow{r} Q$, or $E \vdash P \equiv Q$ with $r = x_0$.

Proof sketch. This is a formal statement of the phenomenon observed earlier in the example (fig. 5): $\llbracket R \rrbracket_p \otimes \text{act}$ cannot be focused on the right unless $P \equiv R$, in which case the derivation ends with no more foci on inter . If not, the only elements available for focus are inter and one of the congruence rules $\llbracket E \rrbracket_e$ in the unrestricted context. In the former case, the derived rule consumes the $\uparrow \text{act}@s$, and by the time act is produced again, its world has advanced to $s \cdot r$. In the latter case, the definition of a top level X_n in ζP is (un)folded (without advancing the world). \square

Lemma 23 is a strong statement about HyLL derivations using this encoding: every partial derivation using the derived inference rules represents a prefix of an $S\pi$ trace. This is sometimes referred to as *full adequacy*, to distinguish it from adequacy proofs that require complete derivations [26]. The structure of focused derivations is crucial because it allows us to close branches early (using init). It is impossible to perform a similar analysis on unfocused proofs for this encoding; both the encoding and the framework will need further features to implement a form of staging [5, Chapter 3].

Corollary 24 (soundness).

If the canonical sequent $\llbracket E \rrbracket_e, \text{rates}, \text{inter}@x_0; \uparrow \text{act}@x_0, \zeta P \rrbracket_{@x_0}; \cdot \Longrightarrow \cdot; (\llbracket Q \rrbracket_p \text{ at } x_0) \otimes \text{act}@r$ is derivable, then $E \vdash P \xrightarrow{r}^ Q$.*

Proof. Directly from lem. 23. \square

4.2 Simulating $S\pi$ in HyLL

So far the $\text{HyLL}(\mathcal{R})$ encoding of $S\pi$ represents any $S\pi$ trace symbolically. However, not every symbolic trace of an $S\pi$ process can be produced according to the operational semantics of $S\pi$. This subsection sketches how to use the Gillespie algorithm directly on the canonical sequents to compute only correct traces. A thorough treatment of the details of such a simulator is out of scope of this paper, although they can be obtained by analogy with the SPiM simulator [28] for $S\pi$.

The Gillespie algorithm answers the question of which of the several competing enabled actions in a canonical sequent to select as the “next” action from the race condition encoded by the sequent. Because of the focusing restriction, these enabled actions are easy to compute. Each element of ζP is of the form $\text{se1} \multimap \llbracket M \rrbracket_s$, so the enabled actions in that element are given precisely by the topmost occurrences of \uparrow in $\llbracket M \rrbracket_s$. Because none of the sums can have any restricted channels (they have all been removed in the active decomposition of the process earlier), the rates of all the channels will be found in the rates component of the canonical sequent. The apparent rate of a channel x is related to its inherent rate by scaling by a factor proportional to the *activity* on the channel, as defined in [28]. Note that this definition is on the *rate constants* of exponential distributions, not the rate functions themselves; it is sufficient for most practical purposes to assume that the rate functions are exponential distributions.

The Gillespie algorithm prescribes that the channel with the highest apparent rate “wins” the race condition. This is exactly equivalent to choosing the instance for the witness for the $\forall r$ in the int and syn components of inter . Once the winning channel has been selected, the choice of sums and the terms of the sum that will interact on that channel is exactly as non-deterministic as the focusing choices that remain for sums involving that channel, which is

already handled by the overall proof search strategy. Note that the rounds of the algorithm do not have an associated *delay* element as in [28]; instead, we compute (symbolically) a distribution over the delays of a sequence of actions. Numerical delay values must be computed by a subsequent Monte-Carlo evaluation of the rate sequence.

The simulator that is so defined is particular to our encoding of the $S\pi$. A different stochastic process calculus will need a different simulation algorithm. The benefit of focused HyLL as a logical framework is that all such simulators can be defined as particular focusing strategies.

5 Related and future work

The combination of linear logic with labelled deduction isn't new to this work. In the η -logic [13] the constraint domain is intervals of time, and the rules of the logic generate constraint inequalities as a side-effect; however its sole aim is the representation of proof-carrying authentication, and it does not deal with genericity or focusing. The main feature of η not in HyLL is a separate constraint context that gives new constrained propositions. HyLL is also related to the Hybrid Logical Framework (HLF) [33] which captures linear logic itself as a labelled form of intuitionistic logic. Encoding constrained π calculi directly in HLF would be an interesting exercise, requiring a combination of linearity and the constraints.

Probably the most well known and relevant stochastic formalism not already discussed is that of stochastic Petri-nets [22], which have a number of sophisticated model checking tools. Closely related are formalisms such as CSL [2], a stochastic extension of CTL, and the Probabilistic CTL (PCTL) [17], for which there are a number of tailored tools including the PRISM framework [21]. In all these formalisms, the emphasis is to find fragments for which the model checking problem is tractable, so they necessarily sacrifice expressiveness in favor of decidability. Because we intend HyLL as a logical framework, we make the opposite tradeoff—indeed, even ordinary propositional linear logic is undecidable—for efficiency and automation are not our primary motivations. Instead, our aim with HyLL is to construct a logic with as natural a proof theory as possible. To our knowledge, there has never been an attempt to define a logic-programming engine or a theorem prover based on stochastic inference rules.

Besides its obvious merits as a logical framework, logics with localized truth and certain hybrid operators have recently seen some renewed interest in the domain of distributed programming languages; for instance, [25] uses a variant of *at* where the worlds are nodes in a network, with the movement between worlds limited by “send” and “receive” protocols. Lambda calculi enriched with stochastic computational monads have also been proposed [32], but their operational semantics is given in terms of repeated sampling instead of symbolic reasoning. To our knowledge, there has never been an attempt to define a logic-programming engine or a theorem prover based on stochastic inference rules.

Of formalisms specially designed for systems biology, there are far too many to compare carefully here. Possibly the most declarative of these formalisms is the κ -calculus [11], where reactions are reductions on graphs with certain state annotations. It appears (though this has not been formalized) that the κ -calculus can be embedded in HyLL even more naturally than $S\pi$, because a solution—a multiset of chemical products—is simply a tensor of all the internal states of the binding sites together with the formed bonds. One important innovation of κ is the ability to extract semantically meaningful “stories” from simulations. We believe that HyLL provides a natural formal language for such stories. There are also low-level formalisms using various kinds of process calculi; besides $S\pi$, one prominent formalism is PEPA [19] that closely resembles $S\pi$ but uses bounded interactions instead of ν . We leave the embedding PEPA in HyLL to future work, noting that we do not foresee any difficulty.

Several other instantiations of HyLL seem interesting. For the temporal encoding, we can already use disjunction (\oplus) to explain disjunctive states, but it is also possible to obtain a more extensional branching by treating the worlds as points in a partially-ordered set. Another possibility is to consider lists of instants instead of individual instants – this allows us to define periodic availability of a resource, such as one being produced by an oscillating process. (Generally, given a monoid homomorphism μ , we get a natural transformation $\uparrow^\mu: \text{HyLL}(-) \rightarrow \text{HyLL}(\mu(-))$.) As already mentioned, another interesting domain is that of probabilities. We conjecture that a very similar style of encoding can be applied to get an adequate encoding of the probabilistic π calculus.

6 Conclusion

We have presented HyLL, a hybrid linear logic with a well established proof-theoretic pedigree, particularly a cut-free focused sequent calculus, and argued for its use as a logical framework for reasoning with constrained state transition systems by giving an adequate encoding of the synchronous stochastic π -calculus. Our next step will be to implement the Gillespie algorithm as outlined in sec. 4.2 directly on this encoding. One obvious question is whether this algorithm can be implemented once to support more than one encoding of stochastic process calculi. As long as the essential device of our encoding—limiting the stochastic updates to a single atomic proposition—is used, such a generic implementation of the Gillespie algorithm is in principle straightforward.

An important open question for future work is whether a general logic such as HyLL can serve as a framework for specialized logics such as CSL and PCTL. A related question is what benefit linearity truly provides for such logics – linearity is obviously crucial for encoding process calculi that are inherently stateful, but CSL has no such notion of single consumption of resources.

Acknowledgements This work was partially supported by INRIA through the “Equipes Associées” Slimmer, by the INRIA-MSR joint project “*Tools for Specifications and Proofs*”, by the Information Society Technologies programme of the European Commission, Future and Emerging Technologies under the IST-2005-015905 MOBIUS project, and by the European TYPES project.

References

- [1] Jean-Marc Andreoli. Logic programming with focusing proofs in linear logic. *J. of Logic and Computation*, 2(3):297–347, 1992.
- [2] A. Aziz, K. Sanwal, V. Singhal, and R. Brayton. Model checking continuous time Markov chains. *ACM Transactions on Computational Logic*, 1(1):162–170, 2000.
- [3] Marco Bozzano. *A Logic-Based Approach to Model Checking of Parameterized and Infinite-State Systems*. PhD thesis, DISI, Università di Genova, 2002.
- [4] Luca Cardelli. Brane calculi. In *Proceedings of BIO-CONCUR’03*, volume 180. Elsevier ENTCS, 2003.
- [5] Iliano Cervesato, Frank Pfenning, David Walker, and Kevin Watkins. A concurrent logical framework II: Examples and applications. Technical Report CMU-CS-02-102, Carnegie Mellon University, 2003. Revised, May 2003.
- [6] Nathalie Chabrier-Rivier, François Fages, and Sylvain Soliman. The biochemical abstract machine BIOCHAM. In *International Workshop on Computational Methods in Systems Biology (CMSB-2)*. Springer-Verlag LNCS, 2004.
- [7] Bor-Yuh Evan Chang, Kaustuv Chaudhuri, and Frank Pfenning. A judgmental analysis of linear logic. Technical Report CMU-CS-03-131R, Carnegie Mellon University, December 2003.
- [8] Kaustuv Chaudhuri. Focusing strategies in the sequent calculus of synthetic connectives. In Iliano Cervesato, Helmut Veith, and Andrei Voronkov, editors, *15th International Conference on Logic, Programming, Artificial Intelligence and Reasoning (LPAR)*, volume 5330 of LNCS, pages 467–481, November 2008.
- [9] Kaustuv Chaudhuri, Frank Pfenning, and Greg Price. A logical characterization of forward and backward chaining in the inverse method. *J. of Automated Reasoning*, 40(2-3):133–177, March 2008.
- [10] Vincent Danos and Jean Krivine. Formal molecular biology done in CCS. In *Proceedings of BIO-CONCUR’03*, volume 180, pages 31–49. Elsevier ENTCS, 2003.
- [11] Vincent Danos and Cosimo Laneve. Formal molecular biology. *Theor. Comput. Sci.*, 325(1):69–110, 2004.
- [12] Josée Desharmais and Prakash Panangaden. Continuous stochastic logic characterizes bisimulation of continuous-time Markov processes. *Journal of Logic and Algebraic Programming*, 56:99–115, 2003.

- [13] Henry DeYoung, Deepak Garg, and Frank Pfenning. An authorization logic with explicit time. In *Computer Security Foundations Symposium (CSF-21)*, pages 133–145. IEEE Computer Society, 2008.
- [14] Michael B. Elowitz and Stanislas Leibler. A synthetic oscillatory network of transcriptional regulators. *Nature*, 403(6767):335–338, 20 January 2000.
- [15] Jean-Yves Girard. Linear logic. *Theoretical Computer Science*, 50:1–102, 1987.
- [16] Jean-Yves. Girard. Linear logic. *Theoretical Computer Science*, 50:1–102, 1987.
- [17] H. Hansson and B. Jonsson. A logic for reasoning about time and probability. *Formal Aspects of Computing*, (6), 1994.
- [18] Jane Hillston. *A compositional approach to performance modelling*. Cambridge University Press, 1996.
- [19] J. Hilston. *A compositional approach to performance modelling*. Cambridge University Press, 1996.
- [20] Johan Anthony Willem Kamp. *Tense Logic and the Theory of Linear Order*. PhD thesis, University of California, Los Angeles, 1968.
- [21] M. Kwiatkowska, G. Norman, and D. Parker. Probabilistic symbolic model checking using PRISM: a hybrid approach. *International Journal of Software Tools for Technology Transfer*, 6(2), 2004.
- [22] M. Ajmone Marsan, G. Balbo, G. Conte, S. Donatelli, and G. Franceschinis. *Modelling with Generalised Stochastic Petri Nets*. Wiley Series in Parallel Computing. Wiley and Sons, 1995.
- [23] Dale Miller. The π -calculus as a theory in linear logic: Preliminary results. In E. Lamma and P. Mello, editors, *3rd Workshop on Extensions to Logic Programming*, number 660 in LNCS, pages 242–265, Bologna, Italy, 1993. Springer-Verlag.
- [24] Robin Milner. *Communicating and Mobile Systems : The π -calculus*. Cambridge University Press, New York, NY, USA, 1999.
- [25] Tom Murphy, Karl Crary, Robert Harper, and Frank Pfenning. A symmetric modal lambda calculus for distributed computing. In *LICS-19*, pages 286–295. IEEE Press, July 2004.
- [26] Vivek Nigam and Dale Miller. Focusing in linear meta-logic. In *Proceedings of IJCAR: International Joint Conference on Automated Reasoning*, volume 5195 of *LNAI*, pages 507–522. Springer, 2008.
- [27] Frank Pfenning and Conal Elliott. Higher-order abstract syntax. In *Proceedings of the ACM-SIGPLAN Conference on Programming Language Design and Implementation*, pages 199–208. ACM Press, June 1988.
- [28] Andrew Phillips and Luca Cardelli. A correct abstract machine for the stochastic pi-calculus. *Concurrent Models in Molecular Biology*, August 2004.
- [29] Andrew Phillips and Luca Cardelli. A correct abstract machine for the stochastic pi-calculus. In *Proceedings of BioConcur'04*, ENTCS, 2004.
- [30] Andrew Phillips, Luca Cardelli, and Giuseppe Castagna. A graphical representation for biological processes in the stochastic pi-calculus. *Transactions on Computational Systems Biology VII*, pages 123–152, 2006.
- [31] Arthur Prior. *Past, Present and Future*. Oxford University Press, 1967.
- [32] Norman Ramsey and Avi Pfeffer. Stochastic lambda calculus and monads of probability distributions. In *29th ACM Symp. on Principles of Programming Languages*, pages 154–165, 2002.
- [33] Jason Reed. Hybridizing a logical framework. In *International Workshop on Hybrid Logic (HyLo)*, Seattle, USA, August 2006.

- [34] A. Regev, E. M. Panina, W. Silverman, L. Cardelli, and E. Shapiro. Bioambients: an abstraction for biological compartments. *Theoretical Computer Science*, 325(1):141–167, 2004.
- [35] A. Regev, W. Silverman, and E. Shapiro. Representation and simulation of biochemical processes using the π -calculus and process algebra. In L. Hunter R. B. Altman, A. K. Dunker and T. E. Klein, editors, *Pacific Symposium on Biocomputing*, volume 6, pages 459–470, Singapore, 2001. World Scientific Press.
- [36] Uluç Saranlı and Frank Pfenning. Using constrained intuitionistic linear logic for hybrid robotic planning problems. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3705–3710. IEEE, 2007.
- [37] Robert J. Simmons and Frank Pfenning. Linear logical algorithms. In Luca Aceto, Ivan Damgård, Leslie Ann Goldberg, Magnús M. Halldórsson, Anna Ingólfssdóttir, and Igor Walukiewicz, editors, *ICALP 2008: 35th International Colloquium Automata, Languages and Programming, Reykjavik, Iceland*, volume 5126 of *LNCS*, pages 336–347. Springer, July 2008.

A Proofs

A.1 Identity principle

Theorem 25 (Identity principle). *The following rule is derivable.*

$$\frac{}{\Gamma; A@w \Rightarrow A@w} \text{init}^*$$

Proof. By induction on the structure of A . We have the following cases.

Case A is an atom $p \vec{t}$. Then, $\Gamma; p \vec{t}@w \Rightarrow p \vec{t}@w$ by *init*.

Case A is $B \& C$.

$$\frac{\frac{\frac{}{\Gamma; B@w \Rightarrow B@w} \text{i.h.}}{\Gamma; B \& C@w \Rightarrow B@w} \&L_1 \quad \frac{\frac{}{\Gamma; C@w \Rightarrow C@w} \text{i.h.}}{\Gamma; B \& C@w \Rightarrow C@w} \&L_2}{\Gamma; B \& C@w \Rightarrow B \& C@w} \&R$$

Case A is \top .

$$\frac{}{\Gamma; \top@w \Rightarrow \top@w} \top R$$

Case A is $B \oplus C$.

$$\frac{\frac{\frac{}{\Gamma; B@w \Rightarrow B@w} \text{i.h.}}{\Gamma; B@w \Rightarrow B \oplus C@w} \oplus R_1 \quad \frac{\frac{}{\Gamma; C@w \Rightarrow C@w} \text{i.h.}}{\Gamma; C@w \Rightarrow B \oplus C@w} \oplus R_2}{\Gamma; B \oplus C@w \Rightarrow B \oplus C@w} \oplus L$$

Case A is $\mathbf{0}$.

$$\frac{}{\Gamma; \mathbf{0}@w \Rightarrow \mathbf{0}@w} \mathbf{0}L$$

Case A is $B \multimap C$.

$$\frac{\frac{\frac{}{\Gamma; B@w \Rightarrow B@w} \text{i.h.}}{\Gamma; B \multimap C@w, B@w \Rightarrow C@w} \multimap L \quad \frac{}{\Gamma; C@w \Rightarrow C@w} \text{i.h.}}{\Gamma; B \multimap C@w \Rightarrow B \multimap C@w} \multimap R$$

Case A is $B \otimes C$.

$$\frac{\frac{\overline{\Gamma; B@w \Rightarrow B@w} \text{ i.h.} \quad \overline{\Gamma; C@w \Rightarrow C@w} \text{ i.h.}}{\overline{\Gamma; B@w, C@w \Rightarrow B \otimes C@w} \otimes R} \otimes L}{\overline{\Gamma; B \otimes C@w \Rightarrow B \otimes C@w} \otimes L} \otimes R$$

Case A is 1 .

$$\frac{\overline{\Gamma; \cdot \Rightarrow 1@w} \text{ 1R}}{\overline{\Gamma; 1@w \Rightarrow 1@w} \text{ 1L}} \text{ 1L}$$

Case A is $\forall x. B$.

$$\frac{\frac{\overline{\Gamma; B@w \Rightarrow B@w} \text{ i.h.}}{\overline{\Gamma; \forall \alpha. B@w \Rightarrow B@w} \forall L} \forall L}{\overline{\Gamma; \forall \alpha. B@w \Rightarrow \forall \alpha. B@w} \forall R^\alpha} \forall R^\alpha$$

Case A is $\exists x. B$.

$$\frac{\frac{\overline{\Gamma; B@w \Rightarrow B@w} \text{ i.h.}}{\overline{\Gamma; B@w \Rightarrow \exists \alpha. B@w} \exists R} \exists R}{\overline{\Gamma; \exists \alpha. B@w \Rightarrow \exists \alpha. B@w} \exists L^\alpha} \exists L^\alpha$$

Case A is $!B$.

$$\frac{\frac{\overline{\Gamma, B@w; B@w \Rightarrow B@w} \text{ i.h.}}{\overline{\Gamma, B@w; \cdot \Rightarrow B@w} \text{ copy}} \text{ copy}}{\frac{\overline{\Gamma, B@w; \cdot \Rightarrow !B@w} \text{ !R}}{\overline{\Gamma; !B@w \Rightarrow !B@w} \text{ !L}} \text{ !L}} \text{ !R}$$

Case A is $\downarrow u. B$.

$$\frac{\frac{\overline{\Gamma; [w/u]B@w \Rightarrow [w/u]B@w} \text{ i.h.}}{\overline{\Gamma; \downarrow u. B@w \Rightarrow [w/u]B@w} \downarrow L} \downarrow L}{\overline{\Gamma; \downarrow u. B@w \Rightarrow \downarrow u. B@w} \downarrow R} \downarrow R$$

Case A is $(B \text{ at } v)$.

$$\frac{\frac{\overline{\Gamma; B@v \Rightarrow B@v} \text{ i.h.}}{\overline{\Gamma; (B \text{ at } v)@w \Rightarrow B@v} \text{ atL}} \text{ atL}}{\overline{\Gamma; (B \text{ at } v)@w \Rightarrow (B \text{ at } v)@w} \text{ atR}} \text{ atR}$$

□

A.2 Cut admissibility

Theorem 26 (Cut admissibility). *The following two rules are admissible.*

$$\frac{\Gamma; \Delta \Rightarrow A@w \quad \Gamma; \Delta', A@w \Rightarrow C@w'}{\Gamma; \Delta, \Delta' \Rightarrow C@w'} \text{ cut}$$

$$\frac{\Gamma; \cdot \Rightarrow A@w \quad \Gamma, A@w; \Delta \Rightarrow C@w'}{\Gamma; \Delta \Rightarrow C@w'} \text{ cut!}$$

Proof. Name the two premise derivations \mathcal{D} and \mathcal{E} respectively. The proof proceeds by induction on the structure of the derivations \mathcal{D} and \mathcal{E} , and more precisely on a lexicographic order that allows the induction hypothesis to be used whenever:

1. The cut formula becomes strictly smaller (in the subformula relation), or
2. The cut formula remains the same, but an instance of cut is used to justify an instance of cut!.
3. The cut formula remains the same, but the derivation \mathcal{D} is strictly smaller, or
4. The cut formula remains the same, but the derivation \mathcal{E} is strictly smaller, or

In each case, we consider derivations to be identical that differ in such a way that one can be derived from the other simply by weakening and contracting the unrestricted contexts of their respective sequents. The lexicographic order is well-founded because the given derivations \mathcal{D} and \mathcal{E} are finite, and cut! is used at most once per subformula of A (see “copy cuts” below). All the cuts break down into the following four major categories.

Atomic cuts where the formula A is an atom $p(\vec{t})$. We have the following two cases;

Case. \mathcal{D} is:

$$\frac{}{\Gamma ; p(\vec{t})@w \Rightarrow p(\vec{t})@w} \text{init}$$

Then the result of the cut has the same conclusion as that of \mathcal{E} .

Case. \mathcal{E} is

$$\frac{}{\Gamma ; p(\vec{t})@w \Rightarrow p(\vec{t})@w} \text{init}$$

Then the result of the cut has the same conclusion as that of \mathcal{D} .

Principal cuts where a non-atomic cut formula A is introduced by a final right rule in \mathcal{D} and a final left-rule in \mathcal{E} . We have the following cases.

Case. A is $A_1 \& A_2$, and:

$$\mathcal{D} = \frac{\mathcal{D}_1 :: \Gamma ; \Delta \Rightarrow A_1@w \quad \mathcal{D}_2 :: \Gamma ; \Delta \Rightarrow A_2@w}{\Gamma ; \Delta \Rightarrow A_1 \& A_2@w} \&R \quad \mathcal{E} = \frac{\mathcal{E}' :: \Gamma ; \Delta', A_i@w \Rightarrow C@w'}{\Gamma ; \Delta', A_1 \& A_2@w \Rightarrow C@w'} \&L_i$$

Then:

$$\Gamma ; \Delta, \Delta' \Rightarrow C@w' \quad \text{cut on } \mathcal{D}_i \text{ and } \mathcal{E}'.$$

Case. A is $A_1 \oplus A_2$, and:

$$\mathcal{D} = \frac{\mathcal{D}' :: \Gamma ; \Delta \Rightarrow A_i@w}{\Gamma ; \Delta \Rightarrow A_1 \oplus A_2@w} \oplus R_i \quad \mathcal{E} = \frac{\mathcal{E}_1 :: \Gamma ; \Delta', A_1@w \Rightarrow C@w' \quad \mathcal{E}_2 :: \Gamma ; \Delta', A_2@w \Rightarrow C@w'}{\Gamma ; \Delta', A_1 \oplus A_2@w \Rightarrow C@w'} \oplus L$$

Then:

$$\Gamma ; \Delta, \Delta' \Rightarrow C@w' \quad \text{cut on } \mathcal{D}' \text{ and } \mathcal{E}_i.$$

Case. A is $A_1 \multimap A_2$, and:

$$\mathcal{D} = \frac{\mathcal{D}' :: \Gamma ; \Delta, A_1@w \Rightarrow A_2@w}{\Gamma ; \Delta \Rightarrow A_1 \multimap A_2@w} \multimap R \quad \mathcal{E} = \frac{\mathcal{E}_1 :: \Gamma ; \Delta'_1 \Rightarrow A_1@w \quad \mathcal{E}_2 :: \Gamma ; \Delta'_2, A_2@w \Rightarrow C@w'}{\Gamma ; \Delta'_1, \Delta'_2, A_1 \multimap A_2 \Rightarrow C@w'} \multimap L$$

Then:

$$\begin{aligned} \Gamma ; \Delta, A_1@w, \Delta'_2 \Rightarrow C@w' & \quad \text{cut on } \mathcal{D}' \text{ and } \mathcal{E}_2. \\ \Gamma ; \Delta, \Delta'_1, \Delta'_2 \Rightarrow C@w' & \quad \text{cut on } \mathcal{E}_1 \text{ and above.} \end{aligned}$$

Case. A is $A_1 \otimes A_2$, and:

$$\mathcal{D} = \frac{\mathcal{D}_1 :: \Gamma; \Delta_1 \Rightarrow A_1 @ w \quad \mathcal{D}_2 :: \Gamma; \Delta_2 \Rightarrow A_2 @ w}{\Gamma; \Delta_1, \Delta_2 \Rightarrow A_1 \otimes A_2 @ w} \otimes R \quad \mathcal{E} = \frac{\mathcal{E}' :: \Gamma; \Delta', A_1 @ w, A_2 @ w \Rightarrow C @ w'}{\Gamma; \Delta', A_1 \otimes A_2 @ w \Rightarrow C @ w'} \otimes L$$

Then:

$$\begin{array}{l} \Gamma; \Delta', \Delta_2, A_1 @ w \Rightarrow C @ w' \\ \Gamma; \Delta', \Delta_1, \Delta_2 \Rightarrow C @ w' \end{array} \quad \begin{array}{l} \text{cut on } \mathcal{D}_2 \text{ and } \mathcal{E}' \\ \text{cut on } \mathcal{D}_1 \text{ and above.} \end{array}$$

Case. A is $\mathbf{1}$, and:

$$\mathcal{D} = \frac{}{\Gamma; \cdot \Rightarrow \mathbf{1} @ w} \mathbf{1}R \quad \mathcal{E} = \frac{\mathcal{E}' :: \Gamma; \Delta' \Rightarrow C @ w'}{\Gamma; \Delta', \mathbf{1} @ w \Rightarrow C @ w'} \mathbf{1}L$$

The result of the cut is the conclusion of \mathcal{E}' .

Case. A is $\forall x. B$, and:

$$\mathcal{D} = \frac{\mathcal{D}'(\alpha) :: \Gamma; \Delta \Rightarrow B @ w}{\Gamma; \Delta \Rightarrow \forall \alpha. B @ w} \forall R^\alpha \quad \mathcal{E} = \frac{\mathcal{E}' :: \Gamma; \Delta', [\tau/\alpha]B @ w \Rightarrow C @ w'}{\Gamma; \Delta', \forall \alpha. B @ w \Rightarrow C @ w'} \forall L$$

Let a be any parameter. Then:

$$\Gamma; \Delta, \Delta' \Rightarrow C @ w' \quad \text{cut on } \mathcal{D}'(\tau) \text{ and } \mathcal{E}'.$$

Case. A is $\exists x. B$, and:

$$\mathcal{D} = \frac{\mathcal{D}' :: \Gamma; \Delta \Rightarrow [\tau/\alpha]B @ w}{\Gamma; \Delta \Rightarrow \exists \alpha. B @ w} \exists R \quad \mathcal{E} = \frac{\mathcal{E}'(\alpha) :: \Gamma; \Delta', B @ w \Rightarrow C @ w'}{\Gamma; \Delta', \exists \alpha. B @ w \Rightarrow C @ w'} \exists L^\alpha$$

Let a be any parameter. Then:

$$\Gamma; \Delta, \Delta' \Rightarrow C @ w' \quad \text{cut on } \mathcal{D}' \text{ and } \mathcal{E}'(\alpha).$$

Case. A is $!B$, and:

$$\mathcal{D} = \frac{\mathcal{D}' :: \Gamma; \cdot \Rightarrow B @ w}{\Gamma; \cdot \Rightarrow !B @ w} !R \quad \mathcal{E} = \frac{\mathcal{E}' :: \Gamma, B @ w; \Delta' \Rightarrow C @ w'}{\Gamma; \Delta', !B @ w \Rightarrow C @ w'} !L$$

Then:

$$\Gamma; \Delta' \Rightarrow C @ w' \quad \text{cut! on } \mathcal{D}' \text{ and } \mathcal{E}'.$$

Case. A is $\downarrow u. B$, and:

$$\mathcal{D} = \frac{\mathcal{D}' :: \Gamma; \Delta \Rightarrow [w/u]B @ w}{\Gamma; \Delta \Rightarrow \downarrow u. B @ w} \downarrow R \quad \mathcal{E} = \frac{\mathcal{E}' :: \Gamma; \Delta', [w/u]B @ w \Rightarrow C @ w'}{\Gamma; \Delta', \downarrow u. B @ w \Rightarrow C @ w'} \downarrow L$$

Then:

$$\Gamma; \Delta, \Delta' \Rightarrow C @ w' \quad \text{cut on } \mathcal{D}' \text{ and } \mathcal{E}'.$$

Case. A is $(B \text{ at } v)$, and:

$$\mathcal{D} = \frac{\mathcal{D}' :: \Gamma; \Delta \Rightarrow B @ v}{\Gamma; \Delta \Rightarrow (B \text{ at } v) @ w} \text{at}R \quad \mathcal{E} = \frac{\mathcal{E}' :: \Gamma; \Delta', B @ v \Rightarrow C @ w'}{\Gamma; \Delta', (B \text{ at } v) @ w \Rightarrow C @ w'} \text{at}L$$

Then:

$$\Gamma; \Delta, \Delta' \Rightarrow C @ w' \quad \text{cut on } \mathcal{D}' \text{ and } \mathcal{E}'.$$

Copy cuts where the cut formula in \mathcal{E} was transferred using copy, i.e.:

$$\mathcal{D} :: \Gamma ; \cdot \Longrightarrow A@w \quad \mathcal{E} = \frac{\mathcal{E}' :: \Gamma, A@w ; \Delta', A@w \Longrightarrow C@w'}{\Gamma, A@w ; \Delta' \Longrightarrow C@w'} \text{ copy}$$

Here,

$$\begin{array}{l} \Gamma, A@w ; \cdot \Longrightarrow A@w \\ \Gamma, A@w ; \Delta' \Longrightarrow C@w' \\ \Gamma ; \Delta' \Longrightarrow C@w' \end{array} \quad \begin{array}{l} \text{weakening on } \mathcal{D}. \\ \text{cut on } \mathcal{D} \text{ and } \mathcal{E}'. \\ \text{cut! on } \mathcal{D} \text{ and above.} \end{array}$$

The first cut is applied on a variant of \mathcal{D} that differs from \mathcal{D} only in terms of a weaker unrestricted context. In the last step, a cut was used to justify a cut!, which is allowed by the lexicographic order.

Left-commutative cuts where the cut formula A is a side formula in the derivation \mathcal{D} . The following is a representative case.

$$\mathcal{D} = \frac{\mathcal{D}' :: \Gamma ; \Delta, D@w'', E@w'' \Longrightarrow A@w}{\Gamma ; \Delta, D \otimes E@w'' \Longrightarrow A@w} \otimes L \quad \mathcal{E} :: \Gamma ; \Delta', A@w \Longrightarrow C@w'.$$

Here,

$$\begin{array}{l} \Gamma ; \Delta, D@w'', E@w'', \Delta' \Longrightarrow C@w' \\ \Gamma ; \Delta, \Delta', D \otimes E@w'' \Longrightarrow C@w' \end{array} \quad \begin{array}{l} \text{cut on } \mathcal{D}' \text{ and } \mathcal{E}. \\ \otimes L. \end{array}$$

Right-commutative cuts where the cut formula A is a side formula in the derivation \mathcal{E} . The following is a representative case.

$$\mathcal{D} :: \Gamma ; \Delta \Longrightarrow A@w \quad \mathcal{E} = \frac{\mathcal{E}_1 :: \Gamma ; \Delta', A@w \Longrightarrow D@w' \quad \mathcal{E}_2 :: \Gamma ; \Delta', A@w \Longrightarrow E@w'}{\Gamma ; \Delta', A@w \Longrightarrow D \& E@w'} \& R$$

Here,

$$\begin{array}{l} \Gamma ; \Delta, \Delta' \Longrightarrow D@w' \\ \Gamma ; \Delta, \Delta' \Longrightarrow E@w' \\ \Gamma ; \Delta, \Delta' \Longrightarrow D \& E@w' \end{array} \quad \begin{array}{l} \text{cut on } \mathcal{D} \text{ and } \mathcal{E}_1. \\ \text{cut on } \mathcal{D} \text{ and } \mathcal{E}_2. \\ \& R. \end{array}$$

This completes the inventory of all possible cuts. □

A.3 Invertibility

Theorem 27 (Invertibility). *The following rules are invertible:*

1. *On the right:* $\&R$, $\top R$, $\neg R$, $\forall R$, $\downarrow R$ and $@R$;
2. *On the left:* $\otimes L$, $\mathbf{1}L$, $\oplus L$, $\mathbf{0}L$, $\exists L$, $!L$, $\downarrow L$ and $\text{at}L$.

Proof. Each inversion is shown to be admissible using a suitable cut.

Case of $\&R$:

$$\frac{\frac{\frac{\Gamma ; \Delta \Longrightarrow A_1 \& A_2@w}{\Gamma ; \Delta \Longrightarrow A_1 \& A_2@w} \&L_i \quad \frac{\Gamma ; A_i@w \Longrightarrow A_i@w}{\Gamma ; A_i@w \Longrightarrow A_i@w} \text{init}^*}{\Gamma ; \Delta \Longrightarrow A_i@w} \text{cut}}{\Gamma ; \Delta \Longrightarrow A_i@w} \&L_i$$

Case of $\top R$: trivial.

Case of $\neg\circ R$:

$$\frac{\frac{\frac{\Gamma; A@w \Rightarrow A@w \text{ init}^*}{\Gamma; A \neg\circ B@w} \quad \frac{\frac{\Gamma; B@w \Rightarrow B@w \text{ init}^*}{\Gamma; A \neg\circ B@w, A@w \Rightarrow B@w} \neg\circ L}{\Gamma; \Delta, A@w \Rightarrow B@w} \text{ cut}}{\Gamma; \Delta \Rightarrow A \neg\circ B@w} \text{ cut}}{\Gamma; \Delta, A@w \Rightarrow B@w} \text{ cut}$$

Case of $\forall R$:

$$\frac{\frac{\frac{\Gamma; A@w \Rightarrow A@w \text{ init}^*}{\Gamma; \Delta \Rightarrow \forall\alpha. A@w} \quad \frac{\frac{\Gamma; \forall\alpha. A@w \Rightarrow A@w \text{ } \forall L}{\Gamma; \Delta \Rightarrow A@w} \text{ cut}}{\Gamma; \Delta \Rightarrow \forall\alpha. A@w} \text{ cut}}{\Gamma; \Delta \Rightarrow A@w} \text{ cut}}{\Gamma; \Delta \Rightarrow A@w} \text{ cut}$$

Case of $\downarrow R$:

$$\frac{\frac{\frac{\Gamma; [w/u]A@w \Rightarrow [w/u]A@w \text{ init}^*}{\Gamma; \Delta \Rightarrow \downarrow u. A@w} \quad \frac{\frac{\Gamma; \downarrow u. A@w \Rightarrow [w/u]A@w \text{ } \downarrow L}{\Gamma; \Delta \Rightarrow [w/u]A@w} \text{ cut}}{\Gamma; \Delta \Rightarrow \downarrow u. A@w} \text{ cut}}{\Gamma; \Delta \Rightarrow [w/u]A@w} \text{ cut}}{\Gamma; \Delta \Rightarrow [w/u]A@w} \text{ cut}$$

Case of $\text{at}R$:

$$\frac{\frac{\frac{\Gamma; A@v \Rightarrow A@v \text{ init}^*}{\Gamma; \Delta \Rightarrow (A \text{ at } v)@w} \quad \frac{\frac{\Gamma; (A \text{ at } v)@w \Rightarrow A@v \text{ } \text{at}L}{\Gamma; \Delta \Rightarrow A@v} \text{ cut}}{\Gamma; \Delta \Rightarrow (A \text{ at } v)@w} \text{ cut}}{\Gamma; \Delta \Rightarrow A@v} \text{ cut}}{\Gamma; \Delta \Rightarrow A@v} \text{ cut}$$

Case of $\otimes L$:

$$\frac{\frac{\frac{\Gamma; A@w \Rightarrow A@w \text{ init}^*}{\Gamma; A@w, B@w \Rightarrow A \otimes B@w} \quad \frac{\frac{\Gamma; B@w \Rightarrow B@w \text{ init}^*}{\Gamma; A@w, B@w \Rightarrow A \otimes B@w} \otimes R}{\Gamma; \Delta, A \otimes B@w \Rightarrow C@w'} \text{ cut}}{\Gamma; \Delta, A@w, B@w \Rightarrow C@w'} \text{ cut}}{\Gamma; \Delta, A@w, B@w \Rightarrow C@w'} \text{ cut}$$

Case of $\mathbf{1}L$:

$$\frac{\frac{\Gamma; \cdot \Rightarrow \mathbf{1}@w \text{ } \mathbf{1}R}{\Gamma; \Delta \Rightarrow C@w'} \quad \Gamma; \Delta, \mathbf{1}@w \Rightarrow C@w'}{\Gamma; \Delta \Rightarrow C@w'} \text{ cut}$$

Case of $\oplus L$:

$$\frac{\frac{\frac{\Gamma; A_i@w \Rightarrow A_i@w \text{ init}^*}{\Gamma; A_i@w \Rightarrow A_1 \oplus A_2@w} \oplus R_i}{\Gamma; \Delta, A_i@w \Rightarrow C@w'} \quad \Gamma; \Delta, A_1 \oplus A_2@w \Rightarrow C@w'}{\Gamma; \Delta, A_i@w \Rightarrow C@w'} \text{ cut}$$

Case of $\mathbf{0}L$: trivial.

Case of $\exists L$:

$$\frac{\frac{\frac{\Gamma; A@w \Rightarrow A@w \text{ init}^*}{\Gamma; A@w \Rightarrow \exists\alpha. A@w} \exists R}{\Gamma; \Delta, A@w \Rightarrow C@w'} \quad \Gamma; \Delta, \exists x. A@w \Rightarrow C@w'}{\Gamma; \Delta, A@w \Rightarrow C@w'} \text{ cut}$$

Case of !L:

$$\frac{\frac{\frac{\Gamma, A@w; A@w \Rightarrow A@w}{\Gamma, A@w; \cdot \Rightarrow A@w} \text{init}^* \text{copy}}{\Gamma, A@w; \cdot \Rightarrow !A@w} !R \quad \frac{\Gamma; \Delta, !A@w \Rightarrow C@w'}{\Gamma, A@w; \Delta, !A@w \Rightarrow C@w'} \text{weaken}}{\Gamma, A@w; \Delta \Rightarrow C@w'} \text{cut}$$

Case of ↓L:

$$\frac{\frac{\frac{\Gamma; [w/u]A@w \Rightarrow [w/u]A@w}{\Gamma; [w/u]A@w \Rightarrow \downarrow u.A@w} \text{init}^* \downarrow R}{\Gamma; \Delta, \downarrow u.A@w \Rightarrow C@w'} \text{cut}}{\Gamma; \Delta, [w/u]A@w \Rightarrow C@w'}}$$

Case of atL:

$$\frac{\frac{\frac{\Gamma; A@v \Rightarrow A@v}{\Gamma; A@v \Rightarrow (A \text{ at } v)@w} \text{init}^* \text{atR}}{\Gamma; \Delta, A@v \Rightarrow C@w'} \text{cut}}{\Gamma; \Delta, (A \text{ at } v)@w \Rightarrow C@w'}}$$

□

A.4 Correctness and consistency

Theorem 28 (Correctness of the sequent calculus).

1. If $\Gamma; \Delta \Rightarrow C@w$, then $\Gamma; \Delta \vdash C@w$. (soundness)
2. If $\Gamma; \Delta \vdash C@w$, then $\Gamma; \Delta \Rightarrow C@w$. (completeness)

Proof. The right rules of the sequent calculus and the introduction rules of natural deduction coincide. Therefore, for (1), we need only to show that the judgemental and left rules of the sequent calculus are admissible in natural deduction, and for (2), only to show that the judgemental and elimination rules of natural deduction are admissible in the sequent calculus. The following are the main cases.

⇒/⊢ case. (init)

$$\frac{}{\Gamma; p(\vec{t})@w \vdash p(\vec{t})@w} \text{hyp}$$

⇒/⊢ case. (copy)

$$\frac{\frac{}{\Gamma, A@w; \cdot \vdash A@w} \text{hyp!} \quad \Gamma, A@w; \Delta, A@w \vdash C@w'}{\Gamma, A@w; \Delta \vdash C@w'} \text{subst}$$

⇒/⊢ case. (&L_i)

$$\frac{\frac{\frac{}{\Gamma; A_1 \& A_2@w \vdash A_1 \& A_2@w} \text{hyp}}{\Gamma; A_1 \& A_2@w \vdash A_i@w} \&E_i \quad \Gamma; \Delta, A_i@w \vdash C@w'}{\Gamma; \Delta, A_1 \& A_2@w \vdash C@w'} \text{subst}$$

⇒/⊢ case. (⊕L)

$$\frac{\frac{}{\Gamma; A_1 \oplus A_2@w \vdash A_1 \oplus A_2@w} \text{hyp} \quad \Gamma; \Delta, A_1@w \vdash C@w' \quad \Gamma; \Delta, A_2@w \vdash C@w'}{\Gamma; \Delta, A_1 \oplus A_2@w \vdash C@w'} \oplus E$$

\Rightarrow /t case. (0L)

$$\frac{\overline{\Gamma; \mathbf{0}_{@w} \vdash \mathbf{0}_{@w}} \text{ hyp}}{\Gamma; \Delta, \mathbf{0}_{@w} \vdash C_{@w'}} \mathbf{0}E$$

\Rightarrow /t case. (\otimes L)

$$\frac{\overline{\Gamma; A \otimes B_{@w} \vdash A \otimes B_{@w}} \text{ hyp} \quad \Gamma; \Delta, A_{@w}, B_{@w} \vdash C_{@w'}}{\Gamma; \Delta, A \otimes B_{@w} \vdash C_{@w'}} \otimes E$$

\Rightarrow /t case. (1L)

$$\frac{\overline{\Gamma; \mathbf{1}_{@w} \vdash \mathbf{1}_{@w}} \text{ hyp} \quad \Gamma; \Delta \vdash C_{@w'}}{\Gamma; \Delta, \mathbf{1}_{@w} \vdash C_{@w'}} \mathbf{1}E$$

\Rightarrow /t case. (\neg O L)

$$\frac{\overline{\Gamma; A \neg B_{@w} \vdash A \neg B_{@w}} \text{ hyp} \quad \Gamma; \Delta \vdash A_{@w} \quad \neg E \quad \Gamma; \Delta', B_{@w} \vdash C_{@w'}}{\Gamma; \Delta, \Delta', A \neg B_{@w} \vdash C_{@w'}} \text{ subst}$$

\Rightarrow /t case. (\forall L)

$$\frac{\overline{\Gamma; \forall \alpha. A_{@w} \vdash \forall \alpha. A_{@w}} \text{ hyp} \quad \Gamma; \forall \alpha. A_{@w} \vdash [\tau/\alpha]A_{@w} \quad \forall E \quad \Gamma; \Delta, [\tau/\alpha]A_{@w} \vdash C_{@w'}}{\Gamma; \Delta, \forall \alpha. A_{@w} \vdash C_{@w'}} \text{ subst}$$

\Rightarrow /t case. (\exists L)

$$\frac{\overline{\Gamma; \exists \alpha. A_{@w} \vdash \exists \alpha. A_{@w}} \text{ hyp} \quad \Gamma; \Delta, A_{@w} \vdash C_{@w'}}{\Gamma; \Delta, \exists \alpha. A_{@w} \vdash C_{@w'}} \exists E^\alpha$$

\Rightarrow /t case. (!L)

$$\frac{\overline{\Gamma; !A_{@w} \vdash !A_{@w}} \text{ hyp} \quad \Gamma, !A_{@w}; \Delta \vdash C_{@w'}}{\Gamma; \Delta, !A_{@w} \vdash C_{@w'}} !E$$

\Rightarrow /t case. (\downarrow L)

$$\frac{\overline{\Gamma; \downarrow u. A_{@w} \vdash \downarrow u. A_{@w}} \text{ hyp} \quad \Gamma; \downarrow u. A_{@w} \vdash [w/u]A_{@w} \quad \downarrow E \quad \Gamma; \Delta, [w/u]A_{@w} \vdash C_{@w'}}{\Gamma; \Delta, \downarrow u. A_{@w} \vdash C_{@w'}} \text{ subst}$$

\Rightarrow /t case. (at L)

$$\frac{\overline{\Gamma; (A \text{ at } v)_{@w} \vdash (A \text{ at } v)_{@w}} \text{ hyp} \quad \Gamma; (A \text{ at } v)_{@w} \vdash A_{@v} \quad \text{at } E \quad \Gamma; \Delta, A_{@v} \vdash C_{@w'}}{\Gamma; \Delta, (A \text{ at } v)_{@w} \vdash C_{@w'}} \text{ subst}$$

$\vdash \Rightarrow$ case. (hyp)

$$\frac{}{\Gamma; A@w \Rightarrow A@w} \text{init}^*$$

$\vdash \Rightarrow$ case. (hyp!)

$$\frac{\frac{}{\Gamma, A@w; A@w \Rightarrow A@w} \text{init}^*}{\Gamma, A@w; \cdot \Rightarrow A@w} \text{copy}}$$

$\vdash \Rightarrow$ case. ($\&E_i$)

$$\frac{\frac{\frac{}{\Gamma; \Delta \Rightarrow A_1 \& A_2@w} \text{init}^*}{\Gamma; A_i@w \Rightarrow A_i@w} \&L_i}{\Gamma; \Delta \Rightarrow A_i@w} \text{cut}}$$

$\vdash \Rightarrow$ case. ($\oplus E$)

$$\frac{\frac{\frac{}{\Gamma; \Delta \Rightarrow A \oplus B@w} \text{init}^*}{\Gamma; \Delta', A@w \Rightarrow C@w'} \oplus L}{\Gamma; \Delta, \Delta' \Rightarrow C@w'} \text{cut}}$$

$\vdash \Rightarrow$ case. ($\mathbf{0}E$)

$$\frac{\frac{}{\Gamma; \Delta \Rightarrow \mathbf{0}@w} \text{init}^*}{\Gamma; \Delta', \mathbf{0}@w \Rightarrow C@w'} \mathbf{0}L}{\Gamma; \Delta, \Delta' \Rightarrow C@w'} \text{cut}}$$

$\vdash \Rightarrow$ case. ($\otimes E$)

$$\frac{\frac{\frac{}{\Gamma; \Delta \Rightarrow A \otimes B@w} \text{init}^*}{\Gamma; \Delta', A@w, B@w \Rightarrow C@w'} \otimes L}{\Gamma; \Delta, \Delta' \Rightarrow C@w'} \text{cut}}$$

$\vdash \Rightarrow$ case. ($\mathbf{1}E$)

$$\frac{\frac{}{\Gamma; \Delta \Rightarrow \mathbf{1}@w} \text{init}^*}{\Gamma; \Delta', \mathbf{1}@w \Rightarrow C@w'} \mathbf{1}L}{\Gamma; \Delta, \Delta' \Rightarrow C@w'} \text{cut}}$$

$\vdash \Rightarrow$ case. ($\forall E$)

$$\frac{\frac{\frac{}{\Gamma; \Delta \Rightarrow \forall \alpha. A@w} \text{init}^*}{\Gamma; [\tau/\alpha]A@w \Rightarrow [\tau/\alpha]A@w} \forall L}{\Gamma; \Delta \Rightarrow [\tau/\alpha]A@w} \text{cut}}$$

$\vdash \Rightarrow$ case. ($\exists E$)

$$\frac{\frac{\frac{}{\Gamma; \Delta \Rightarrow \exists \alpha. A@w} \text{init}^*}{\Gamma; \Delta', A@w \Rightarrow C@w'} \exists L^\alpha}{\Gamma; \Delta, \Delta' \Rightarrow C@w'} \text{cut}}$$

$\vdash \Rightarrow$ case. ($\mathbf{!}E$)

$$\frac{\frac{}{\Gamma; \Delta \Rightarrow \mathbf{!}A@w} \text{init}^*}{\Gamma; \Delta', \mathbf{!}A@w \Rightarrow C@w'} \mathbf{!}L}{\Gamma; \Delta, \Delta' \Rightarrow C@w'} \text{cut}}$$

$\vdash \Rightarrow$ case. ($\downarrow E$)

$$\frac{\frac{\Gamma; \Delta \Rightarrow \downarrow u. A@w \quad \frac{\overline{\Gamma; \Delta', [w/u]A@w \Rightarrow [w/u]A@w} \text{ hyp}}{\Gamma; \Delta', \downarrow u. A@w \Rightarrow [w/u]A@w} \downarrow L}{\Gamma; \Delta, \Delta' \Rightarrow [w/u]A@w} \text{ cut}}{\Gamma; \Delta, \Delta' \Rightarrow [w/u]A@w} \text{ cut}$$

$\vdash \Rightarrow$ case. (at E)

$$\frac{\Gamma; \Delta \Rightarrow (A \text{ at } v)@w \quad \frac{\overline{\Gamma; A@v \Rightarrow A@v} \text{ init}^*}{\Gamma; (A \text{ at } v)@w \Rightarrow A@v} \text{ at } L}{\Gamma; \Delta \Rightarrow A@v} \text{ cut}$$

□

Corollary 29 (Consistency of HyLL). *There is no proof of $\cdot; \cdot \vdash \mathbf{0}@w$.*

Proof. Suppose $\cdot; \cdot \vdash \mathbf{0}@w$ is derivable. Then, by the completeness and cut-admissibility theorems on the sequent calculus, $\cdot; \cdot \Rightarrow \mathbf{0}@w$ must have a cut-free proof. But, we can see by simple inspection that there can be no cut-free proof of $\cdot; \cdot \Rightarrow \mathbf{0}@w$, as this sequent cannot be the conclusion of any rule of inference in the sequent calculus. Therefore, $\cdot; \cdot \vdash \mathbf{0}@w$ is not derivable. □

A.5 Connection to IS5

Theorem 30 (HyLL is intuitionistic S5). *The following sequent is derivable: $\cdot; \diamond A@w \Rightarrow \square \diamond A@w$.*

Proof.

$$\frac{\frac{\frac{\overline{\cdot; A@a \Rightarrow A@a} \text{ init}^*}{\cdot; A@a \Rightarrow (A \text{ at } a) \text{ at } b} \text{ at } R}{\cdot; A@a \Rightarrow \exists v. (A \text{ at } v)@b} \exists R}{\cdot; (A \text{ at } a)@w \Rightarrow (\exists v. (A \text{ at } v) \text{ at } b)@w} \text{ at } L, \text{ at } R}{\cdot; (A \text{ at } a)@w \Rightarrow \forall u. (\exists v. (A \text{ at } v) \text{ at } u)@w} \forall R^b}{\cdot; \exists u. (A \text{ at } u)@w \Rightarrow \forall u. (\exists v. (A \text{ at } v) \text{ at } u)@w} \exists L^a}{\cdot; \diamond A@w \Rightarrow \square \diamond A@w} \text{ defn}$$

□