Proceedings of Machine Learning Research vol 284:1–24, 2025 19th Conference on Neurosymbolic Learning and Reasoning

mULLER: A Modular Categorical Semantics of the Neurosymbolic ULLER Framework via Monads

Author names withheld

Editor: Under Review for NeSy 2025

Abstract

ULLER (Unified Language for LEarning and Reasoning) provides a single first-order logic (FOL) syntax whose knowledge bases can be used without modification by a broad range of neurosymbolic systems. [1] The original specification endows this syntax with three pairwise independent semantics—classical, fuzzy, and probabilistic—each accompanied by dedicated semantic rules. We prove that these seemingly disparate semantics are all instances of one categorical framework based on *monads*, the very construct that models side effects in functional programming. This enables the *modular* addition of new semantics and systematic translations between them. As examples , we outline the addition of LTN, STL, possibilistic and continuous fuzzy semantics and their relations. In particular, our approach allows a modular implementation of ULLER in Python and Haskell. As a further outcome, we generalize quantification in Logic Tensor Networks (LTN) to arbitrary (also infinite) domains by extending the Giry monad to probability spaces.[2] EdNote(1) EdNote(2)

1. Introduction

Neurosymbolic integration is a rapidly developing branch of AI. In the past, numerous heterogeneous approaches have emerged, each with its own code base. [11] introduces ULLER, a unified neurosymbolic library that aspires to play for neurosymbolic systems the role that TensorFlow and PyTorch play for deep-learning workflows. Their theoretical core is the concept of a NeSy system: standard first-order logic enriched with neural components. In particular, formulas of the form

$$x := m(T_1, \dots, T_n)(F)$$

are used to integrate neural models m into logical formulas. These models leave the platonic logical world of objects and truth values. Instead, they perform computations that may return multiple values and typically involve non-determinism or probability distributions as illustrated in their following toy example:

$$x := dice()(x = 6 \land even(x))$$

throws a die once, and checks whether it is 6 and even (so the resulting probability is $\frac{1}{6}$), whereas

$$x := dice()(x = 6) \land x := dice()(even(x))$$

[3] throws a die twice, and checks whether the first one is 6 and the second one is even EdNote(3)

1. EdNote: **Daniel**: Introduce the word MULLER here: noun, a heavy tool of stone or iron (usually with a flat base and a handle) that is used to grind and mix material.
2. EdNote: **1Fh6**: "The abstract claims it 'proves' that several semantics are instances of their categorical framework. However proofs of these equivalences are missing."
3. EdNote: **3Pt1**: "If you add computational predicates, how do these fit within the scoping concerns of ULLER? For example, winLottery() ∧ winLottery() are two independent events according to your semantics?

(so the resulting probability is $\frac{1}{12}$). With neural classifiers, regressions or other machine learning models in place of the *dice*() function, one obtains a neurosymbolic first-order logic. However, the notion of NeSy system in [11] has several shortcomings:

- There is no uniform inductive definition of truth, i.e. of the truth value of a sentence in an interpretation. Rather, the notion of NeSy system has the inductive interpretation function as a component, meaning that classical, probabilistic and fuzzy NeSy systems employ three different inductive definitions of truth. Parts of these definitions of truth are copied verbatim from one NeSy system to another, other parts need to be replaced. This duplication of semantic rules is not modular. By contrast, we aim at a truly uniform inductive definition of truth value that is independent of the NeSy system and hence can be reused for different NeSy systems, such that the NeSy system itself is a parameter of the inductive definition of truth.

- The case of continuous probability distributions (involving probability kernels or Markov kernels) is not covered faithfully, because in this case, measurable spaces are required to properly define the mentioned Markov kernels. However, measurable spaces are not considered in [11].

- The treatment of logical connectives is not uniform across NeSy systems, i.e. the different sets of connectives are not considered as instances of a common abstract (algebraic) notion. Moreover, predicates are only two-valued and not e.g. fuzzy. Also, quantifiers in probabilistic semantics are defined using possibly infinite products, without requiring a suitable order structure on domains and without discussing convergence.

- The high-level concepts of semantics and computation are not properly separated. Computation (namely sampling) is mixed into the semantics at least in two places. The first one in the classical semantics, where the possibly multi-valued arg max can only be properly evaluated using sampling. We conceptualize the arg max differently as a transition *between* semantics. The second time is in "Sampling Semantics", which is not really semantics but computation (sampling).

We argue that ULLER is conceptually robust and show that a monadic formulation resolves all of the foregoing issues. In particular, ULLER formulas of the form

$$x := m(T_1, \ldots, T_n)(F)$$

can be modeled using Moggi's notion of computational monad [7], which has been introduced to model side effects in (functional) programming. Although monads originate in category theory, we always begin with a concrete presentation over the category **Set** (that can be understood without any knowledge of category theory) and only then sketch the generalization to an arbitrary cartesian closed category. [4]

EdNote(4)

How to have them as one (cf the first dice example in the introduction)? I thought non-determinism is allowed only in statements to be more transparent on this."

4. EdNote: **3Pt1**: "Add more intuitions and concrete examples: While the paper states to "always begin with a concrete presentation over the category that can be understood without any knowledge of category thery" (btw, the notation for the category of sets is never introduced), the examples that follow quickly delve into abstract notations without developing intuitions. A few words around each concept, and a more concrete example with an actual program, its inputs and outputs, would make this a lot clearer."

2. Neurosymbolic frameworks

A neurosymbolic framework (NeSy framework) is the general framework for a system that combines neural models with symbolic logic, and that provides the semantic background for the specific logic involved. Examples are the logics behind DeepProbLog [6] or Logic Tensor Networks [2]. The notion of NeSy framework is not defined in [11]. Rather, they define a notion of NeSy system, which is quite ad-hoc, because it makes a fundamental choice about the logic, while simultaneously being based on a particular interpretation, i.e. a particular choice of a neural and logical model. Our notion of NeSy framework provides a means to disentangle these choices. A particular NeSy framework can be used to define different NeSy systems, while [11] needs to individually bake in the framework into each system, causing semantic rule duplication.

2.1. Monads

We interpret formulas $x := m(T_1, \ldots, T_n)(F)$ involving neural models m (=certain computations) using Moggi's notion of computational monad [7] in the form of Kleisli triples:

Definition 1 *A* ***set-based Kleisli triple*** $(\mathcal{T}, \eta, (-)^*)$ *consists of:*

- *A mapping $\mathcal{T}$ mapping a set X to the set $\mathcal{T}X$ (of computations with values from X),*
- *A family of functions: $\eta_X : X \to \mathcal{T}X$ for each set X (construing a value $a \in X$ as stateless computation $\eta_X(a) \in \mathcal{T}X$),*
- *A function that assigns to each function $f : X \to \mathcal{T}Y$ a function $f^* : \mathcal{T}X \to \mathcal{T}Y$ (called the* Kleisli extension*), needed for sequential composition of computations*[1]*,*

such that the following axioms hold:

1. $(\eta_X)^* = \mathrm{id}_{\mathcal{T}X}$,
2. $f^* \circ \eta_X = f$ *for all* $f : X \to \mathcal{T}Y$,
3. $(g^* \circ f)^* = g^* \circ f^*$ *for all* $f : X \to \mathcal{T}Y$ *and* $g : Y \to \mathcal{T}Z$.

Example 1 ***Probability distribution monad (Kleisli triple)*** $\mathcal{D}$ ***on* Set.**

$$\mathcal{D}X := \Big\{ \rho : X \to [0,1] \;\Big|\; \rho \textit{ has finite support and } \sum_{x \in X} \rho(x) = 1 \Big\},$$

$$\eta_X(x) := \delta_x,\ \delta_x(y) = \begin{cases} 1, & x = y \\ 0, & x \neq y \end{cases} \quad f^*(\rho)(y) := \sum_{x \in X} \rho(x) f(x)(y) \quad \big(f : X \to \mathcal{D}Y,\ \rho \in \mathcal{D}X\big).$$

5 EdNote(5)

1. Namely, $f : X \to \mathcal{T}Y$ and $g : Y \to \mathcal{T}Z$ can be composed to $g^* \circ f : X \to \mathcal{T}Z$.
5. EdNote:

Categorical generalisation: A Kleisli triple on a category $\mathcal{C}$ involves a mapping of objects $\mathcal{T} : \mathrm{Ob}(\mathcal{C}) \to \mathrm{Ob}(\mathcal{C})$, a family of morphisms $\eta_X : X \to \mathcal{T}X$ for each object X in $\mathcal{C}$ (called the *unit*), and a function that assigns to each morphism $f : X \to \mathcal{T}Y$ a morphism $f^* : \mathcal{T}X \to \mathcal{T}Y$ such that the axioms hold as in Def. 1. Note that a set-based Kleisli triple is then a Kleisli triple on the category **Set**.

Example 2 ***Giry monad (Kleisli triple) $\mathcal{G}$ on* Meas*, the category of measurable spaces and maps.*** *For a measurable space (X, Σ_X)*

$$\mathcal{G}(X, \Sigma_X) := \big\{\textit{probability measures } \rho \textit{ on } (X, \Sigma_X)\big\},$$

$$\eta_{(X,\Sigma_X)}(x) := \delta_x, \quad f^*(\rho)(A) := \int\limits_X f(x)(A) d\rho(x) \quad \big(f : (X, \Sigma_X) \to \mathcal{G}(Y, \Sigma_Y), A \subseteq X \textit{ measurable}\big).$$

2.2. Double Commutative Monoid Lattices

We need an algebraic structure to model the space of truth values. We weaken the notion of BL algebra [4] used in fuzzy logic as follows:

Definition 2 (Double Commutative Monoid Lattice (2CMon-Lat)) *A* double commutative monoid lattice[2] *is a tuple*

$$\big(S,\ 0,\ 1,\ \leq,\ \otimes,\ \oplus,\ \to\big)$$

in which S is a set and $\mathcal{L} := (S, \leq, 0, 1)$ forms a bounded lattice. We have the smallest element $0 \in S$ *and* largest element $1 \in S$*. The other operations are given as the maps:*

$$\otimes,\ \oplus,\ \to :\ S \times S \longrightarrow S.$$

and form two commutative monoids $\big(S,\ \otimes,\ 1\big)$ and $\big(S,\ \oplus,\ 0\big)$.

For quantification, we want to allow for different aggregation operations other than infinite meet and join to cover the quantifiers of Logic Tensor Networks [2].

Definition 3 (Aggregated 2CMon-Lat) *An* aggregated 2CMon-Lat *has a pair of order preserving maps, one pair of maps for each set X:*

$$\mathrm{aggr}^{\forall}_X, \mathrm{aggr}^{\exists}_X :\ \mathcal{L}^X \longrightarrow \mathcal{L}.$$

In the case of a complete lattice, $\mathrm{aggr}^{\forall}_X$ can be chosen as meet $\bigwedge_X$ and $\mathrm{aggr}^{\exists}_X$ as join $\bigvee_X$.

Categorical generalisation: *A complete 2CMon-Lat internal to a cartesian closed category $\mathcal{C}$[3] consists of an object A in $\mathcal{C}$, a lattice on A internal to $\mathcal{C}$, morphisms $\oplus, \otimes, \to : A \times A \to A$, $0, 1 : 1_{\mathcal{C}} \to A$ and for each object $X \in \mathrm{Ob}(\mathcal{C})$ morphisms $\mathrm{aggr}^{\forall}_X, \mathrm{aggr}^{\exists}_X : A^X \to A$, such that the above axioms hold when appropriately interpreted in $\mathcal{C}$[4].*

2. This is a generalization of residual lattices to also include BL algebras, Heyting Algebras, DeMorgan Algebras and Continuous Semirings. It is inspired by residual lattices as in [4] and also by quantales. Logician's note: in accordance with the NeSy literature, we will not differentiate between weak and strong connectives; however, this could easily be adapted.
3. Such a category has exponential objects ("function spaces") A^B and currying, that is, for $f : A \times B \to C$, there is $\Lambda(f) : A \to C^B$.
4. $1_{\mathcal{C}}$ is a terminal object.

2.3. Neurosymbolic frameworks

Given some basic notion of truth Ω, we assume that NeSy systems work on the monadic space of truth values $\mathcal{T}\Omega$, which is required to be a 2CMon-Lat. If $\mathcal{T}$ is the identity monad, $\mathcal{T}\{0,1\}$ is just the two-element set $\{0,1\}$ of classical truth values. If $\mathcal{T}$ is the distribution or probability monad, $\mathcal{T}\{0,1\}$ is the unit interval $[0,1]$, which can be regarded as the space of probabilistic or fuzzy truth values.

Definition 4 (NeSy framework) *A* NeSy framework $(\mathcal{T},\mathcal{R})$ *consists of*

1. *a strong monad $\mathcal{T}$ with strength $\mathcal{S}$ on a cartesian closed category $\mathcal{C}$,*[5]
2. *an aggregated 2CMon-Lat $\mathcal{R}$ internal to $\mathcal{C}$ on $\mathcal{T}\Omega$ for some $\Omega \in \mathrm{Ob}_{\mathcal{C}}$ of truth values.*

Examples are given in Table 1 and discussed in more detail in section 4.

Table 1: NeSy Framework Examples

Logic/Theory	$\mathcal{C}$	$\mathcal{T}$	Ω	$\mathcal{T}\Omega$	**2CMon-Lat**
Classical	**Set**	Identity	$\{0,1\}$	$\{0,1\}$	Boolean Alg.
Three-valued LP	**Set**	Powerset $\mathcal{P}_{\neq\emptyset}$	$\{0,1\}$	$\{0,B,1\}$	Kleene/Priest Alg.
Classical Fuzzy	**Set**	Identity	$[0,1]$	$[0,1]$	BL–Alg.
Discrete Probabilistic	**Set**	Distribution $\mathcal{D}$	$\{0,1\}$	$[0,1]$	Product BL-Alg.
Cont. Probabilistic	**QBS**	Giry $\mathcal{G}$	$\{0,1\}$	$[0,1]$	Product BL-Alg.
Infinitary LTN$_p$	**QProb**	Probability $\mathcal{O}$	$\{0,1\}$	$[0,1]$	Product BL-Alg.

Note that further examples arise by varying the 2CMon-Lat $\mathcal{R}$ on $[0,1]$.

3. Syntax and Semantics of Monadic ULLER

3.1. Syntax of First-Order Logic

The ULLER language of [11] features computational function symbols that can be realized e.g. by neural networks. In a similar spirit, we here add computational predicate symbols, which are also realized by neural networks, for example in Logic Tensor Networks [2].

Definition 5 (NeSy System Signature) *A* NeSy system signature Σ *consists of a set S of* sorts, *two disjoint $S^*{\times}S$-sorted families:* $\mathrm{F} = (\mathrm{F}_{w,s})_{w\in S^*, s\in S}$ *and* $\mathrm{MF} = (\mathrm{MF}_{w,s})_{w\in S^*, s\in S}$ *of* function symbols *and* computational function symbols, *and two disjoint S^*-sorted families:* $\mathrm{P} = (\mathrm{P}_w)_{w\in S^*}$ *and* $\mathrm{MP} = (\mathrm{MP}_w)_{w\in S^*}$ *of* predicate symbols *and* computational predicate symbols.

*Function symbols with no arguments are called constants (*Const*), and (computational) predicate symbols with no arguments are called (computational) propositional symbols (*Prps *and* MPrps *respectively). Function symbols with one argument are called properties* (Prop).

5. TODO: explain strength. Note that **Set** is a cartesian closed category, and every monad on **Set** is strong. We need cartesian closure for the semantics of quantification and computational function symbols.

Concerning syntax, we largely follow the definitions given in [11]. That is, just like in the original ULLER framework we define our syntax as a context-free grammar. Given a *signature* Σ of non-logical symbols and set of variables V, we can define the syntax of first-order logic (FOL) formulas over Σ and V as follows:

Compound Formulas:

$F ::= x := m(T, \ldots, T)(F)$ $\quad [m \in \mathrm{MF}]$

$F ::= \forall x : s\ (F) \mid \exists x : s\ (F)$

$F ::= F \wedge F \mid F \vee F \mid F \rightarrow\ F \mid \neg F \mid (F)$

Atomic Formulas:

$F ::= M(T, \ldots, T) \mid N$ $\quad [M \in \mathrm{MP}, N \in \mathrm{MPrps}]$

$F ::= P(T, \ldots, T) \mid R \mid \bot \mid \top$ $\quad [P \in \mathrm{P}, R \in \mathrm{Prps}]$

Terms:

$T ::= x : s$ $\quad [x \in \mathrm{V}, s \in S]$

$T ::= f(T, \ldots, T) \mid T.\mathsf{prop} \mid c$ $\quad [f \in \mathrm{F}, \mathsf{prop} \in \mathrm{Prop}, c \in \mathrm{Const}]$

3.2. Tarskian Semantics

EdNote(6)

Definition 6 ($\mathcal{T}$-interpretation of a NeSy system signature) [6] *Let $(\mathcal{T}, \mathcal{R})$ be a NeSy framework of a monad on* **Set** *and let Σ be a signature. A $\mathcal{T}$-interpretation $\mathcal{I}$ is given by*

- *a set $\mathcal{I}(s)$ for every sort s,*
- *a function $\mathcal{I}(f) : \mathcal{I}(s_1) \times \ldots \times \mathcal{I}(s_n) \rightarrow \mathcal{I}(s)$ for every (normal) function symbol $f : s_1, \ldots, s_n \rightarrow s$,*
- *a function $\mathcal{I}(m) : \mathcal{I}(s_1) \times \ldots \times \mathcal{I}(s_n) \rightarrow \mathcal{T}(\mathcal{I}(s))$ for every computational function symbol $m : s_1, \ldots, s_n \rightarrow s$,*
- *a function $\mathcal{I}(P) : \mathcal{I}(s_1) \times \ldots \times \mathcal{I}(s_n) \rightarrow \Omega$ for every predicate symbol $P : s_1, \ldots, s_n$,*
- *and a function $\mathcal{I}(M) : \mathcal{I}(s_1) \times \ldots \times \mathcal{I}(s_n) \rightarrow \mathcal{T}\Omega$ for every computational predicate symbol $M : s_1, \ldots, s_n$.* [7] [8]

EdNote(7)
EdNote(8)

In the categorical generalisation, sets are replaced by objects in $\mathcal{C}$ and functions are replaced by morphisms in $\mathcal{C}$. Therefore, a NeSy system *can be defined as a triple $(\mathcal{T}, \mathcal{R}, \mathcal{I})$, where $(\mathcal{T}, \mathcal{R})$ is a NeSy framework and $\mathcal{I}$ is a $\mathcal{T}$-interpretation.*

6. EdNote: **1Fh6**: "Definition 6: The main motivation for the original ULLER was reuse of the same syntacticly specified theory + neural networks. However, if I understand correctly, the different semantics in Table 1 require difference interpretations of the computational symbols, since it depends on the choice of . Doesn't this reduce the reusability?"
7. EdNote: **1Fh6**: "There are two methods for using neural networks (in functions and in predicates), which are handled quite differently. If I understand correctly, this results in behaviour that differs from the standard understanding of the frameworks discussed."
8. EdNote: **1Fh6**: "As mentioned, the original ULLER did not have computational predicate symbols. Do I understand correctly the regular differentiable fuzzy logic behaviour comes from the computational predicate symbols?"

In [11], based on an interpretation, the notion of NeSy system provides a Tarskian inductive definition $[\![\cdot]\!]$ of the semantics of formulas and thus it implicitly also defines the semantics of the logical symbols. The drawback of this approach is that the Tarskian semantics $[\![\cdot]\!]$ is inherently tied to the specific interpretation of the NeSy system. We can disentangle matters here, because we first give a semantics of the logical symbols via a NeSy framework, and based on that, the interpretation provides the semantics of the non-logical symbols. Based on this foundation, the inductive definition of the Tarskian semantics $[\![\cdot]\!]$ needs to be given only once, and is valid across all NeSy frameworks and systems.

Definition 7 (Tarskian semantics $[\![\cdot]\!]$ of formulas) *Given a NeSy framework $(\mathcal{T}, \mathcal{R})$, a* NeSy interpretation $\mathcal{I}$ *over $(\mathcal{T}, \mathcal{R})$ we can determine the interpretation morphisms:*

$$\textbf{\textit{Formulas:}}\ [\![F]\!]_{\mathcal{I}} : \mathcal{V}_F \to \mathcal{T}\Omega, \qquad \textbf{\textit{Terms:}}\ [\![T]\!]_{\mathcal{I}} : \mathcal{V}_T \to \mathcal{I}(s_T).$$

*We define $\mathcal{V}_T := \prod_{x:t\in\Gamma_T} \mathcal{I}(t)$. Here Γ_T is the context of T and s_T is the (unique) sort of the term T. Analogously $\mathcal{V}_F := \prod_{x:t\in\Gamma_F} \mathcal{I}(t)$. Note that if T_1 is a subterm of T_2, there is a projection $\pi_{T_1,T_2} : \mathcal{V}_{T_2} \to \mathcal{V}_{T_1}$, and analogously for formulas. $\boldsymbol{T}$ stands for $T_1, \ldots, T_n$. Moreover, $\langle [\![\boldsymbol{T}]\!]_i \circ \pi_i \rangle_i = \langle [\![T_1]\!] \circ \pi_1, \ldots, [\![T_n]\!] \circ \pi_n \rangle$ and $[\![\boldsymbol{T}]\!] = ([\![T_1]\!], \ldots, [\![T_n]\!])$. Also, recall currying in a cartesian closed category, that is, for $f : A \times B \to C$, there is $\Lambda(f) : A \to C^B$. The categorical semantics ensures that all involved and resulting functions are morphisms in $\mathcal{C}$, i.e. are measurable in case that $\mathcal{C} =$ **Meas**, etc. With a purely set-theoretic semantics, we would need to prove measurability (or other properties) separately for each NeSy framework. That said, besides the general categorical case, for better understandability, we also translate the equations to their meaning in the category of sets. We work with variable valuations $\nu \in \mathcal{V}_T$ (and $\nu \in \mathcal{V}_F$), noting that elements of $\prod_{x:t\in\Gamma_T} \mathcal{I}(t)$ map variables $x : t$ to values in $\mathcal{I}(t)$. We write $[\![T]\!]_{\mathcal{I},\nu} = [\![T]\!]_{\mathcal{I}}(\nu)$ and $[\![F]\!]_{\mathcal{I},\nu} = [\![F]\!]_{\mathcal{I}}(\nu)$. That said, we mostly omit $\mathcal{I}$ and ν if clear from the context.*

4. Examples of Monad Semantics

4.1. Classical and Three-valued Semantics

The classical semantics is simply given by the identity monad on the category of sets and the Boolean algebra on $\Omega = \{0, 1\}$ which results in classical first-order logic.

While the classical semantics is deterministic, for modeling argmax, which occurs in the classical semantics of [11], we will need non-determinism in section 4.3. The (non-empty) powerset monad models non-deterministic computations, cf. multialgebras [12]. These result in non-deterministic truth values. The 2CMon-Lat is that for paraconsistent three-valued logic with $\mathcal{T}\Omega = \{F, T, B\}$, where B stands for *both true and false*. The resulting logic is Priest's Logic of Paradox (LP) [8]. While Priest has introduced his logic for dealing with inconsistent information, the combination with non-deterministic terms arising here quite naturally has not been studied in the literature.

4.2. Probabilistic/Distributional Semantics

Probabilistic Semantics [9] The interpretation of a function f of arity n is a *Markov* EdNote(9)

9. EdNote: **1Fh6**: "For probabilistic semantics I would expect a weighted model count when using computational predicate symbols (which output distributions over true/false). However, the probabilities seem

Table 2: Inductive definition of the Tarskian semantics

Syntax	**Categorical Semantics** $[\![\cdot]\!]_{\mathcal{I}}$	**Set Semantics** $[\![\cdot]\!]_{\mathcal{I},\nu}$
Terms		
$[\![x : s]\!]$	$\mathrm{id}_{\mathcal{I}(s)}$	$\nu(x)$
$[\![f(\boldsymbol{T})]\!]$	$\mathcal{I}(f) \circ \langle [\![\boldsymbol{T}]\!]_i \circ \pi_i \rangle_i$	$\mathcal{I}(f)\big([\![\boldsymbol{T}]\!]\big)$
$[\![T.\mathsf{prop}]\!]$	$\mathcal{I}(\mathsf{prop}) \circ [\![T]\!]$	$\mathcal{I}(\mathsf{prop})([\![T]\!])$
$[\![c]\!]$	$\mathcal{I}(c)$	$\mathcal{I}(c)$
Atomic formulas		
$[\![M(\boldsymbol{T})]\!]$	$\mathcal{I}(M) \circ \langle [\![\boldsymbol{T}]\!]_i \circ \pi_i \rangle_i$	$\mathcal{I}(M)\big([\![\boldsymbol{T}]\!]\big)$
$[\![R(\boldsymbol{T})]\!]$	$\eta_\Omega \circ \mathcal{I}(R) \circ \langle [\![\boldsymbol{T}]\!]_i \circ \pi_i \rangle_i$	$\eta_\Omega\big(\mathcal{I}(R)([\![\boldsymbol{T}]\!])\big)$
$[\![N]\!]$	$\mathcal{I}(N)$	$\mathcal{I}(N)$
$[\![P]\!]$	$\eta_\Omega \circ \mathcal{I}(P)$	$\eta_\Omega(\mathcal{I}(P))$
Compound formulas		
$[\![x := m(\boldsymbol{T})(F)]\!]$	$[\![F]\!]^* \circ \mathcal{S} \circ \big\langle \pi_{V_{F\setminus x:s}}, \mathcal{I}(m) \circ \langle [\![\boldsymbol{T}]\!]_i \circ \pi_i \rangle_i \big\rangle$	$(\lambda a.[\![F]\!]_{\nu[x \mapsto a]})^*\big(I(m)([\![\boldsymbol{T}]\!])\big)$
$[\![\forall x{:}s\, F]\!]$	$\mathrm{aggr}^{\forall}_{\mathcal{I}(s)} \circ \Lambda_s([\![F]\!]) \circ \pi_{V_{F\setminus x:s}}$	$\mathrm{aggr}^{\forall}_{\mathcal{I}(s)}(\lambda a.[\![F]\!]_{\nu[x\mapsto a]})$
$[\![\exists x{:}s\, F]\!]$	$\mathrm{aggr}^{\exists}_{\mathcal{I}(s)} \circ \Lambda_s([\![F]\!]) \circ \pi_{V_{F\setminus x:s}}$	$\mathrm{aggr}^{\exists}_{\mathcal{I}(s)}(\lambda a.[\![F]\!]_{\nu[x\mapsto a]})$
$[\![F \wedge G]\!]$	$\otimes_{\mathcal{R}} \circ \langle [\![F]\!] \circ \pi_F, [\![G]\!] \circ \pi_G \rangle$	$[\![F]\!] \otimes_{\mathcal{R}} [\![G]\!]$
$[\![F \vee G]\!]$	$\oplus_{\mathcal{R}} \circ \langle [\![F]\!] \circ \pi_F, [\![G]\!] \circ \pi_G \rangle$	$[\![F]\!] \oplus_{\mathcal{R}} [\![G]\!]$
$[\![F \to G]\!]$	$\to_{\mathcal{R}} \circ \langle [\![F]\!] \circ \pi_F, [\![G]\!] \circ \pi_G \rangle$	$[\![F]\!] \to_{\mathcal{R}} [\![G]\!]$
$[\![\neg F]\!]$	$[\![F \to \bot]\!]$	$[\![F]\!] \to_{\mathcal{R}} 0_{\mathcal{R}}$
$[\![\top]\!], [\![\bot]\!]$	$1_{\mathcal{R}}, 0_{\mathcal{R}}$	$1_{\mathcal{R}}, 0_{\mathcal{R}}$

kernel, which is a measurable map $X \xrightarrow{q} \mathcal{G}(Y)$ where $\mathcal{G}$ denotes the *Giry monad* on the category of measurable spaces denoted **Meas**. However, **Meas** is not cartesian closed, so we have to generalize to the category **QBS**[6] of quasi-borel spaces. For simplicity of presentation, we will however mostly work within its full subcategory **SMeas** of standard

to be combined using the product algebra - So the probabilistic semantics actually acts as a fuzzy logic when using computational predicate symbols. The only way to introduce weighted model counting is via the computational function symbols. I don't think this would be intuitive for a standard NeSy person."

6. This category is defined and proven to be cartesian closed in [5] and there it's also proven that **SMeas** is a full subcategory of **QBS**.

measurable spaces. We evaluate in the Prodcut BL-Algebra on $[0,1]$ to obtain:

$$[\![x := m(\boldsymbol{T})(F)]\!] := \int_{a\in\mathcal{I}(s_m)} [\![F]\!]_{\nu[x\mapsto a]}\, d\rho_m(\,\cdot \mid \boldsymbol{T})(a) \tag{1}$$

$$[\![\forall x{:}s\ F]\!] := \inf_{a\in\mathcal{I}(s)} [\![F]\!]_{\nu[x\mapsto a]}, \quad [\![\exists x{:}s\ F]\!] := \sup_{a\in\mathcal{I}(s)} [\![F]\!]_{\nu[x\mapsto a]} \tag{2}$$

$$[\![F \wedge G]\!] := [\![F]\!] \cdot [\![G]\!], \quad [\![F \vee G]\!] := [\![F]\!] + [\![G]\!] - [\![F]\!] \cdot [\![G]\!] \tag{3}$$

$$[\![F \to G]\!] := 1 - [\![F]\!] + [\![F]\!] \cdot [\![G]\!], \quad [\![\neg F]\!] := 1 - [\![F]\!] \tag{4}$$

$$[\![\bot]\!] := 0, \quad [\![\top]\!] := 1. \tag{5}$$

Infinitary LTN$_p$ Semantics The infinitary LTN$_p$ semantics is a modification of the probabilistic semantics [10], which is motivated by replacing the quantifiers in equation (2) with the p-means in Stable product real logic of Logic Tensor Networks [2]. The hyperparameter p is usually increased during training, because this moves from mean (tolerant to outliers) towards maximum (logically stricter). We generalize this such that for any sort s we have to provide a probability measure ρ_s on $\mathcal{I}(s)$ to obtain a p-means for infinite domains: EdNote(10)

$$M_p(a_1,\dots,a_n) := \Big(\frac{1}{n}\sum_{i=1}^{n} a_i^p\Big)^{1/p}, \quad M_p(f;\rho_s) := \Big(\int_{x\in X} f(x)^p\, d\rho_s(x)\Big)^{1/p}, \tag{6}$$

and these extend to[7] $M_0(a_1,\dots,a_n) := \Big(\prod_{i=1}^{n} a_i\Big)^{\frac{1}{n}}$ and $M_0(f;\rho_s) := \exp\big(\int \ln f\, d\rho_s\big)$. For $p \to \infty$ we recover the supremum. Take $X = \{1,\dots,N\}$ with counting measure $1/N$, then $M_p(f;\rho_s)$ reduces to $M_p(a_1,\dots,a_n)$ or choose weights w_i summing up to 1 for $i = 1,\dots,N$ to obtain the weighted p-mean. The aggregated 2CMon-Lat is the same as in probabilistic logic, except for the aggregation functions: For a hyperparameter $0 < p < \infty$ of LTN$_p$, let $\mathrm{aggr}^{\exists}_{\mathcal{I}(s)}(f) := M_p(f;\rho_s)$ and $\mathrm{aggr}^{\forall}_{\mathcal{I}(s)}(f) := 1 - M_p(\lambda x.f(1-x);\rho_s)$. As a result, equation (2) now becomes:

$$[\![\exists x{:}s\ F]\!] = \Big(\int_{a\in\mathcal{I}(s)} \big([\![F]\!]_{\nu[x\mapsto a]}\big)^p\, d\rho_s\Big)^{1/p}, \quad [\![\forall x{:}s\ F]\!] = 1 - \Big(\int_{a\in\mathcal{I}(s)} \big([\![F]\!]_{\nu[x\mapsto 1-a]}\big)^p\, d\rho_s\Big)^{1/p} \tag{7}$$

It's worth noting that our probability measure ρ_s depend on the sort s of the variable x in the quantifier, in contrast to [9], where the probability measure depends directly on the variable x. Instead, we extend the category **QBS** to a category **QProb** of quasi-Borel spaces equipped with a probability measure[8] on quasi-Borel spaces and also extend the Giry monad on **QBS** to a monad on **QProb**.

10. EdNote: **1Fh6**: "The infinitary LTN Semantics is a modification of the probabilistic semantics. This is confusing to me, LTN is a fuzzy logic framework, not a probabilistic one. "

7. Here the product is similar to the infinitary product used in ULLER for the semantics of the universal quantifier in probabilistic semantics. Since that infinitary product is however not well-defined, we use $M_0(f;\rho_s)$, which also works for the infinite case and is p-means in the finite case (for a uniform distribution). If we want to obtain a finite product, we alternatively can iterate conjunction.

8. As defined in [5] [p. 3].

Distributional Semantics The Giry monad $\mathcal{G}$ restricts to the *distribution monad* on **Set**. This semantics is the probabilistic semantics of section 4.2 with $\mathcal{G}$ replaced by $\mathcal{D}$, the Giry integral collapses to a finite sum:

$$[\![x := m(\boldsymbol{T})(F)]\!](h) \;=\; \sum_{u \in \mathcal{I}(s_m)} [\![F\backslash x : s]\!](h)(u) \cdot \rho_m(u \mid \boldsymbol{T}_h).$$

4.3. NeSy Framework Shifts

Definition 8 (NeSy Framework Shift) *For any NeSy framework* $(\mathcal{T}, \mathcal{R})$*, let* $\operatorname{Intp}((\mathcal{T}, \mathcal{R}), \Sigma)$ *denote the set of all interpretations over signature* Σ*. A* NeSy framework shift $\gamma : (\mathcal{T}, \mathcal{R}) \to (\mathcal{T}', \mathcal{R}')$ *between two NeSy frameworks* $(\mathcal{T}, \mathcal{R})$ *and* $(\mathcal{T}', \mathcal{R}')$ *is a family of functions* $\operatorname{Intp}(\gamma)_\Sigma : \operatorname{Intp}((\mathcal{T}, \mathcal{R}), \Sigma) \to \operatorname{Intp}((\mathcal{T}', \mathcal{R}'), \Sigma)$ *and we normally just write* $\gamma(\mathcal{I})$ *for* $\operatorname{Intp}(\gamma)_\Sigma(\mathcal{I})$*.*

Argmax shift: From distributional to non-deterministic semantics For a given distributional interpretation $\mathcal{I}(m)$ of a computational function symbol m, we can define a non-deterministic interpretation $\gamma(\mathcal{I})(m)$ of m by defining, where s_m is the sort of m:

$$\gamma(\mathcal{I})(m) := \operatorname*{arg\,max}_{u \in \mathcal{I}(s_m)} \mathcal{I}(m)(u),$$

and for a computational predicate symbol M with a projection $\mathrm{pro} : [0,1] \to \{F, T, B\}$:

$$\gamma(\mathcal{I})(M) := \mathrm{pro} \circ \mathcal{I}(M), \qquad \mathrm{pro}(x) := \begin{cases} T & \text{if } x > 1/2, \\ B & \text{if } x = 1/2, \\ F & \text{if } x < 1/2, \end{cases}$$

and for all other symbols set $\gamma(\mathcal{I}) := \mathcal{I}$. This definition is *not* possible in the general probabilistic case, because probability measures often return zero on *all* single values. It is also a non-deterministic interpretation since it returns a *set* of values instead of a single value. The resulting semantics is three-valued, as in section 4.1. Then, in the practical implementation of ULLER, we can use random sampling (see section 4.4) over a uniform distrubtion (unif) to obtain a single value from the set of values returned by $\gamma(\mathcal{I})(m)$. This gives a precise foundation for the use of $\arg\max$ in the classical semantics in [11].

NeSy shifts diagram This $\arg\max$ shift is just one example of many possible NeSy shifts, some of which we will display in the following *commutative* diagram, at least the shifts of their computational function symbols:

$$\begin{tikzcd}
\mathcal{P}_{\neq\emptyset}(X)\text{(3-valued)} \arrow[dr, hook, "\text{(unif)}"] & & \mathcal{D}(X)\text{(Distributional)} \arrow[ll, two heads, "\arg\max"'] \arrow[dl, hook] & \\
& \mathcal{G}(X)\text{(Probablistic)} \arrow[rr, hook, two heads, "(id)"] & & \mathcal{O}(X)\text{(LTN)} \\
\mathrm{Id}_{\textbf{Set}}(X)\text{(Classical)} \arrow[uu, hook, "\{\cdot\}"] \arrow[ur, hook, "\delta"] \arrow[rr, hook, two heads, "id"] & & \mathrm{Id}_{\textbf{Set}}(X)\text{(Fuzzy)} \arrow[ul, hook, "\delta"'] \arrow[uu, hook, "\delta"'] &
\end{tikzcd}$$

4.4. Sampling

While [11] have introduced a sampling semantics, we think that the semantics should define probabilities, while an implementation can work with e.g. Monte Carlo sampling in order to obtain an approximation that is easier to implement (and, in the case of quantification over infinite domains, unavoidable). Hence, we do not discuss sampling semantics here. But a Monte Carlo convergence theorem can easily be stated and proved.

5. Conclusion

The ULLER language [11] aims at a unifying foundation for neurosymbolic systems. In this paper, we have developed a new semantics for ULLER, based on Moggi's formalisation of computational effects as monads. In contrast to the original semantics, our semantics is truly modular. It is based on a notion of NeSy framework that provides the structure of the computational effects and the space of truth values. This modularity will enable a cleaner, more modular implementation of ULLER in Python [11] and an easier integration of new frameworks, as well as a structured method of translating between different frameworks. Note that the distributional and probabilistic NeSy frameworks will make parameterized interpretations in the sense of [11] differentiable and that this can be integrated by using a category of differentiable manifolds and functions. EdNote(11)

Our work suggests an analogy between ULLER's formulas $x := m(T_1, \ldots, T_n)(F)$ and Haskell's do-notation $\mathbf{do}\ x \leftarrow m(T_1, \ldots, T_n); F$ for computational effects. Inspired by this analogy, one could extend ULLER to a language with computational terms and formulas that may be nested.

11. EdNote: **My3J**: "The paper would benefit from implementation examples or at least sketches demonstrating how the modular semantics could be used in practice, e.g., Python library implementation, and how existing common neurosymbolic systems can be unified under the framework."

References

[1] Krassimir T. Atanassov. Intuitionistic fuzzy sets. *Fuzzy Sets and Systems*, 20(1):87–96, August 1986.

[2] Samy Badreddine, Artur S. d'Avila Garcez, Luciano Serafini, and Michael Spranger. Logic tensor networks. *Artif. Intell.*, 303:103649, 2022.

[3] Didier Dubois, J Lang, and Henri Prade. *Possibilistic logic*, pages 439–513. 01 1994.

[4] Petr Hájek. *Metamathematics of Fuzzy Logic.* Trends in Logic. Springer Netherlands, Dordrecht, 1998.

[5] Chris Heunen, Ohad Kammar, Sam Staton, and Hongseok Yang. A Convenient Category for Higher-Order Probability Theory. In *2017 32nd Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 1–12, June 2017.

[6] Robin Manhaeve, Sebastijan Dumancic, Angelika Kimmig, Thomas Demeester, and Luc De Raedt. Neural probabilistic logic programming in deepproblog. *Artif. Intell.*, 298:103504, 2021.

[7] Eugenio Moggi. Notions of computation and monads. *Information and Computation*, 93(1):55–92, July 1991.

[8] Graham Priest. *An introduction to non-classical logic: From if to is.* Cambridge University Press, 2008.

[9] Natalia Ślusarz, Ekaterina Komendantskaya, Matthew L. Daggitt, Robert Stewart, and Kathrin Stark. Logic of Differentiable Logics: Towards a Uniform Semantics of DL, October 2023.

[10] M Sugeno and T Murofushi. Pseudo-additive measures and integrals. *Journal of Mathematical Analysis and Applications*, 122(1):197–222, February 1987.

[11] Emile Van Krieken, Samy Badreddine, Robin Manhaeve, and Eleonora Giunchiglia. ULLER: A Unified Language for Learning and Reasoning. In Tarek R. Besold, Artur d'Avila Garcez, Ernesto Jimenez-Ruiz, Roberto Confalonieri, Pranava Madhyastha, and Benedikt Wagner, editors, *Neural-Symbolic Learning and Reasoning*, volume 14979, pages 219–239. Springer Nature Switzerland, Cham, 2024.

[12] Michal Walicki and Sigurd Meldal. Multialgebras, power algebras and complete calculi of identities and inclusions. In *Workshop on the Specification of Abstract Data Types*, pages 453–468. Springer, 1994.

Appendix A. General note on Supplementary Material

This appendix is supplementary material in the sense of the call for papers. The paper is self-contained and does not depend on the appendix. The appendix contains some more technical details that are not needed for understanding the paper, but which might be useful for the reader who wants to understand the details of the semantics. We also present more sample NeSy frameworks and their shifts.

Appendix B. Probabilistic Semantics Calculation and Theory

Calculation of the interpretation in probabilistic semantics: Set $H := (x := m(T_1, \dots, T_n)(F))$ and write $\boldsymbol{T} := (T_1, \dots, T_n)$. Fix a valuation $h \in \mathcal{V}_H$. Also write $\boldsymbol{T}_h := (\boldsymbol{T}_{h,1}, \dots, \boldsymbol{T}_{h,n})$ in which $\boldsymbol{T}_{h,i} := [\![T_i]\!] \circ \pi_i(h)$ for $i = 1, \dots, n$. Put $[m(\boldsymbol{T})](h) := \mathcal{I}(m)(\boldsymbol{T}_{h,1}, \dots, \boldsymbol{T}_{h,n}) = \rho_m(\,\cdot \mid \boldsymbol{T}_h)$, which is a (quasi) probability measure on $\mathcal{I}(s_m)$, where s_m is the sort of the function symbol m. Now calculate:

$$\begin{aligned}
[\![x := m(\boldsymbol{T})(F)]\!](h) &= [\![F]\!]^* \circ \mathcal{S}\big(\pi_{V_{F\setminus x:m}}(h),\ [m(\boldsymbol{T})](h)\big) \\
&= [\![F]\!]^*\big(\delta_{\pi_{V_{F\setminus x:m}}(h)} \otimes [m(\boldsymbol{T})](h)\big) \\
&= \int [\![F]\!]\ d\big(\delta_{\pi_{V_{F\setminus x:m}}(h)} \otimes [m(\boldsymbol{T})](h)\big) \\
&= \int_{u \in \mathcal{I}(s_m)} [\![F]\!](\pi_{V_{F\setminus x:m}}(h), u)\ d\big([m(\boldsymbol{T})](h)\big)(u) \\
&= \int_{u \in \mathcal{I}(s_m)} [\![F]\!](\pi_{V_{F\setminus x:m}}(h), u)\ d\rho_m(\,\cdot \mid \boldsymbol{T}_h)(u)
\end{aligned}$$

If we further generalize our logic we can also choose a continuous semiring on $[0, 1]$ or even a continuous field (a continuous semiring which is also a field) on $\mathbb{R}$ in the probabilistic semantics. This is in order to closely resemble the canonical measure-theoretical equations underlying probability theory. That is, view $[\![\_]\!]$ as a probability measure itself and regard F and G as measurable sets of a σ-algebra on a set $\mathcal{X}$.

Appendix C. Logic Tensor Networks Background

Setting Stable product real logic of Logic Tensor Networks [2] uses p-means for finite quantification. The hyperparameter p is usually increased during training, because this moves from mean (tolerant to outliers) towards maximum[9] (logically stricter). However, since domains are generally infinite, we also need to aggregate infinite many truth–scores $(x_i)_{i \in I} \subseteq [0, 1]$. The power–mean extends from the finite case to an *integral* form that is well defined whenever the data are L^p-integrable. Let $(X, \mathcal{A}, \rho)$ be a probability space and $f : X \to [0, 1] \subseteq \mathbb{R}$ a measurable map with $\int_X f\, d\rho \leq 1$. Because $0 \leq f \leq 1$, one automatically has $f \in L^p(\rho)$ for every real p, so p-means are always defined. This bounded–by–one assumption reflects the fact that in our logical reading a truth-score never exceeds 1.

9. This holds for existential quantification. Universal quantification $\forall$ is defined through $\neg\exists x{:}s\neg$, and then the move is towards minimum.

We will concentrate again on the full subcategory **SMeas** and extend it to the category **SProb** of probability spaces of standard Borel spaces and get an interpretation as:

- a probability space (X_s, ρ_s) for every sort s,
- a measure-preserving function $\mathcal{I}(f)$ for every function symbol f,
- a measure-preserving function $\mathcal{I}(m)$ with codomain $\mathcal{O}((X_{s_m}, \rho_{s_m}))$ for every computational function symbol m and its type s_m,
- a measure-preserving function $\mathcal{I}(R)\big($codomain $\{0,1\}\big)$ for every predicate symbol P,
- and a measure-preserving function $\mathcal{I}(M)$ with codomain $\mathcal{O}(\{0,1\}) \cong [0,1]$ for every computational predicate symbol M.

Now we are ready to define our probability monad $\mathcal{O}$ on the category **SProb**, which basically takes a probability space (X, ρ) and returns the measurable space $(\mathcal{G}(X), \rho^{\eta})$, that adds no new information to the probability measure ρ. That's exactly what we want, since we we will only use the probability measure ρ to define the semantics of the formulas, and not the derived probability measure ρ^{η}:

Proposition 9 (and definition of the probability monad $\mathcal{O}$) *We can define a probability monad $\mathcal{O}$ as a monad*[10] *on the category* **SProb** *with η, μ being the unit and multiplication of the Giry monad $\mathcal{G}$ on* **SMeas***:*

$$\begin{aligned} \mathcal{O}((X,\rho)) &:= (\mathcal{G}(X), \rho^{\eta}), \\ \rho^{\eta} &:= B \longmapsto \rho(\eta^{-1}(B)), \ \textit{for } B \subseteq \mathcal{G}(X) \ \textit{measurable}, \\ \eta^{\mathcal{O}} &:= \eta, \quad \mu^{\mathcal{O}} := \mu. \end{aligned}$$

Proof We only need to check whether $\eta^{\mathcal{O}}, \mu^{\mathcal{O}}$ are measure preserving, which follows for the unit by definition:

$$\eta^{\mathcal{O}} : (X, \rho) \ \longrightarrow \ \big(\mathcal{G}(X),\, \rho^{\eta}\big)$$

$$\rho\big((\eta^{\mathcal{O}})^{-1}(B)\big) \ = \ \rho\big(\eta^{-1}(B)\big) \ = \ \rho^{\eta}(B).$$

Now, for the multiplication we have:

$$\mu^{\mathcal{O}} : \big(\mathcal{G}^2(X),\, \rho\circ\eta^{-1}\circ\eta_{\mathcal{G}}^{-1}\big) \ \longrightarrow \ \big(\mathcal{G}(X),\, \rho\circ\eta^{-1}\big)$$

since

$$O^2(X) \ = \ O\big((\mathcal{G}(X), \rho^{\eta})\big) \ = \ \big(\mathcal{G}^2(X),\, (\rho^{\eta})^{\eta}\big),$$

and that means we can write for any measurable set $A \subseteq \mathcal{G}^2(X)$:

$$(\rho^{\eta})^{\eta}(A) \ = \ \rho^{\eta}\big(\eta_{\mathcal{G}}^{-1}(A)\big) \ = \ \rho\big(\eta^{-1} \circ \eta_{\mathcal{G}}^{-1}(A)\big).$$

10. Here we use the traditional category theoretical definition of monad as $(\mathcal{T}, \eta, \mu)$, where μ is monad multiplication. This is however equivalent to the definition as Kleisli triple.

This means we show the measure-preserving property as follows

$$\rho\circ\eta\circ\eta_{\mathcal{G}}^{-1}\big(\mu^{-1}(B)\big) \;=\; \rho\circ\eta^{-1}\big(\eta_{\mathcal{G}}^{-1}(\mu^{-1}(B))\big) \;=\; \rho\circ\eta^{-1}(B),$$

because we know the following fact from the monad laws:

$$A = \eta_{\mathcal{G}}^{-1}\big(\mu^{-1}(B)\big) \iff A = \mu\big(\eta_{\mathcal{G}}(A)\big) = B.$$

■

Appendix D. Possibilistic Semantics and STL

Our final goal in this section is to define monads suitable for the semantics of *Possibility Logic* as in [3] and *STL* (Signal Temporal Logic) as in [9]. First we expand the distribution monad $\mathcal{D}$ from example 1 to the sub-distribution monad $\mathcal{S}$ and then use this monad $\mathcal{S}$ to define a possibility monad $\tilde{\Pi}$, which in turn extends differentiably to a STL$_\phi$ monad $\tilde{\Pi}_\phi$. These four distributional monads are depicted in Table 3. The lengthy and technical proofs that these are indeed monads are omitted here, but will be part of a technical report accompanying this paper.

Table 3: Distributional NeSy Frameworks

Logic/Theory	$\mathcal{C}$	$\mathcal{T}$	Ω	$\mathcal{T}\Omega$	**Alg. Struct.**
Discrete Probabilistic	**Set**	Distribution $\mathcal{D}$	$\{0,1\}$	$[0,1]$	Cont. Semirings
Intuitionistic Fuzzy	**Set**	Sub-Distribution $\mathcal{S}$	$\{0,1\}$	Δ_1	Cont. Semirings
Possibility	**Set**	Possibility Monad $\tilde{\Pi}$	$\{0,1\}$	$[-1,1]$	min / max Alg.
STL$_\phi$	**Set**	STL$_\phi$ Monad $\tilde{\Pi}_\phi$	$\{0,1\}$	$[-\infty,\infty]$	min / max Alg.

D.1. Possibilistic Semantics

D.1.1. Sub-distribution Monad

The sub-distribution monad $\mathcal{S}$ is similar to the distribution monad $\mathcal{D}$ but it allows for finitely supported measures that do not sum up to 1, that means:

Definition 10 (Sub-Distribution Monad $\mathcal{S}$)

$$\mathcal{S}X := \Big\{ p : X \to [0,1] \;\Big|\; p \textit{ has finite support and } \sum_{x\in X} p(x) \le 1 \Big\},$$

$$\eta_X(x) := \delta_x, \delta_x(y) = \begin{cases} 1, & x = y \\ 0, & x \neq y \end{cases} \quad f^*(p)(y) := \sum_{x\in X} p(x) f(x)(y) \quad \big(f : X \to \mathcal{S}Y,\ p \in \mathcal{S}X\big).$$

Lemma 11 *For the two-element set $\Omega = \{0,1\}$ the sub-distribution monad yields*

$$\mathcal{S}\Omega \;\cong\; \Delta_1 \;:=\; \big\{(p_0,p_1) \in [0,1]^2 \mid p_0 + p_1 \le 1\big\}.$$

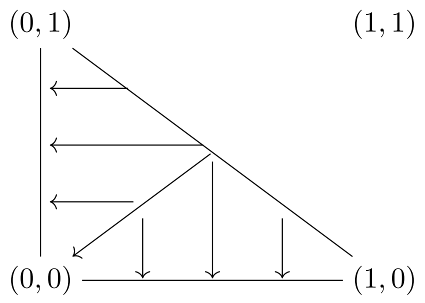
Explicitly, the bijection $\varphi\colon \mathcal{S}\Omega \longrightarrow \Delta_1$ is $\varphi(p) = (p(0), p(1))$ and its inverse sends a pair $(a,b) \in \Delta_1$ to the finitely supported measure $p_{a,b} := a\,\delta_0 + b\,\delta_1$.

Proof A finitely supported $\mathcal{S}$-measure on a finite set is entirely determined by its values on the points of the set. Hence $p \mapsto (p(0), p(1))$ is injective. Because $p(\Omega) = p(0) + p(1) \leq 1$, its image lies in Δ_1. Conversely, every $(a, b) \in \Delta_1$ defines a measure $p_{a,b}$ with total mass $a + b \leq 1$ and finite support $\{0, 1\}$. Thus φ is a bijection. ■

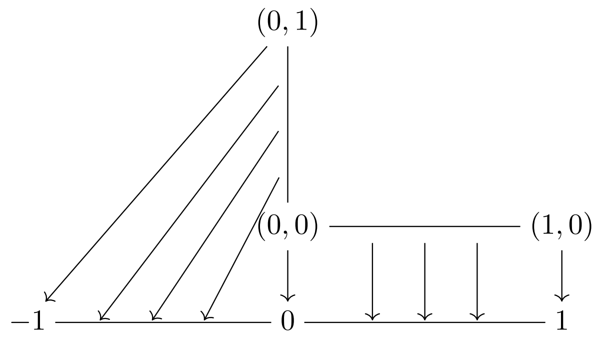
Therefore, as already seen in Table 3, the sub-distribution monad $\mathcal{S}$ and its so-called "Intuitionistic Fuzzy Set Logic"[11] work on the truth value space Δ_1, which we want to *avoid* because it is two-dimensional and not easily relatable to the classical truth values $\{0, 1\}$ and fuzzy truth values $[0, 1]$. Instead, we want to project these down to the one-dimensional space $[-1, 1]$ of possibility values, which we will characterize as the space of truth values of *Possibility Logic* as in [3].

D.1.2. Possibility Monad $\tilde{\Pi}$

Intuition The intuition behind the possibility monad $\tilde{\Pi}$ that we will define is the following: Take the triangle $\mathcal{S}\Omega \cong \Delta_1$ of sub-distributions inside the square and first project it down to the L shaped axes as in the following picture:



And then take the resulting L shape and project it down to the line $[-1, 1]$ as in:



The above pictures can be relabeled in order to match the definitions of the possibility of truth $\Pi(U)$ and the possibility of falsity $\Pi(U^c)$ of an event U[12]:

11. As found in [1].
12. As for example in as for example in [3] [Example 4].

$$\begin{tikzcd}
(0,1) \arrow[r, "\Pi(U^c)"] & (1,1) \\
(0,0) & (1,0) \arrow[u, "\Pi(U)"']
\end{tikzcd}$$

In this, a tuple $(a, b) \in \Delta_1$ is given by the values of $(\Pi(U^c), \Pi(U))$. Also, the necessity of an event is defined as $N(U) := 1 - \Pi(U^c)$. Only values inside, or adjacent to the two arrows are obtained, which follows from the definition of possibility measure. In order to make this coherent with the projection picture and also compatible with fuzzy logic in which $[0, 1]$ are degrees of *truth* and with STL in which $[0, \infty]$ are degrees of *truth* and $[-\infty, 0]$ are degrees of *falsity*, we look at it the following way. The necessity of truth $N(U)$ is interpreted as a degree of *truth* in $[0, 1]$ and the *impossibility* of truth $1 - \Pi(U)$ is interpreted as a degree of *falsity* in $[-1, 0]$. This gives the following picture of values for $(N(U), 1 - \Pi(U))$:

$$\begin{tikzcd}
(0,1) & (1,1) & & & \\
& & -1 & 0 \arrow[l, "-(1-\Pi(U))"'] \arrow[r, "N(U)"] & 1 \\
(0,0) \arrow[uu, "1-\Pi(U)"] \arrow[r, "N(U)"'] & (1,0) & & &
\end{tikzcd}$$

Projecting the L onto the real line Let

$$\mathsf{L} \ := \ \big\{(a, 0) \mid 0 \le a \le 1\big\} \ \cup \ \big\{(0, b) \mid 0 \le b \le 1\big\} \ \subseteq \ \Delta_1$$

be the L–shaped subset of Δ_1 that runs along the two coordinate axes. Define the (piece wise linear) *projection*

$$\varphi : \mathsf{L} \longrightarrow [-1, 1], \qquad \varphi(a, 0) := a, \qquad \varphi(0, b) := -\,b \quad (0 \le a, b \le 1).$$

Hence

$$\varphi(0, 1) = -1, \qquad \varphi(0, 0) = 0, \qquad \varphi(1, 0) = 1,$$

so φ folds the two legs of L onto the interval $[-1, 1]$ in a continuous, order-preserving way.

Min–max algebra on $[-1, 1]$ Define the 2CMon-Lat $\mathcal{R}$ as *idempotent commutative semiring* (a min–max algebra):

$$S \ := \ [-1, 1], \qquad x \otimes_{\mathcal{R}} y \ := \ \max\{x, y\}, \qquad x \oplus_{\mathcal{R}} y \ := \ \min\{x, y\},$$

$$0_{\mathcal{R}} := -1, \qquad 1_{\mathcal{R}} := 1.$$

Back-translation to possibility values For every event U write

$$n := N(U) \in [0,1], \qquad \vartheta := -\big(1 - \Pi(U)\big) \in [-1,0].$$

Because $(n, \vartheta) \in \mathsf{L}$, their projection $\varphi(n, \vartheta) \in [-1,1]$ is exactly the element on which the algebra above acts. The ordering

$$-1 \;<\; \cdots \;<\; 0 \;<\; \cdots \;<\; 1$$

coincides with the intuitive scale "certainly false $\rightarrow$ no information $\rightarrow$ certainly true", so $\oplus_{\mathcal{R}}$ picks the *more plausible* value and $\otimes_{\mathcal{R}}$ the *less plausible* one—just as the usual max / min rules of possibility theory.

Until now we have only worked in the truth value space $[-1,1]$ and explained that we want our monad $\tilde{\Pi}$ to send basic truth values $\Omega = \{0,1\}$ to possibility values $\tilde{\Pi}(\Omega) \cong \mathsf{L} \cong [-1,1]$. In order to formulate a monad however we need to specify what our monad does with arbitrary sets X. The following definition does exactly that, while preserving the above intuition that $\tilde{\Pi}(\Omega) \cong \mathsf{L} \cong [-1,1]$:

Definition 12 (Possibility Monad $\tilde{\Pi}$) *$\tilde{\Pi}$ is defined as follows for sets X:*

$$A_p := \arg\max(p(x)), \qquad \tilde{\pi}(p) := \max(p)\, \delta_{A_p},$$

$$\tilde{\Pi}(X) := \big\{\tilde{\pi}(p) |\, p \in \mathcal{S}(X) : |A_p| = 1\big\} \cup \big\{0\big\} \subseteq \mathcal{S}X,$$
$$\eta_X(x) := \delta_x, \qquad k^*(\tilde{\pi}(p)) := \tilde{\pi}\big(\max(p) k(A_p)\big).$$

D.1.3. STL$_\phi$ Monad

Table 4: Smooth embeddings $\phi : [-1,1] \to \overline{\mathbb{R}}$

name	formula $\phi(x)$	inverse $\phi^{-1}(y)$
scaled tangent	$\tan\!\big(\frac{\pi}{2}x\big)$	$\frac{2}{\pi} \arctan y$
logit	$\ln \dfrac{1+x}{1-x}$	$\dfrac{e^y - 1}{e^y + 1}$
artanh	$\operatorname{artanh}(x)$	$\tanh y$
rational blow–up	$\dfrac{x}{1-x^2}$	$\dfrac{y}{1+\sqrt{1+y^2}}$
arcsine–tan	$\dfrac{x}{\sqrt{1-x^2}}$	$\dfrac{y}{\sqrt{1+y^2}}$

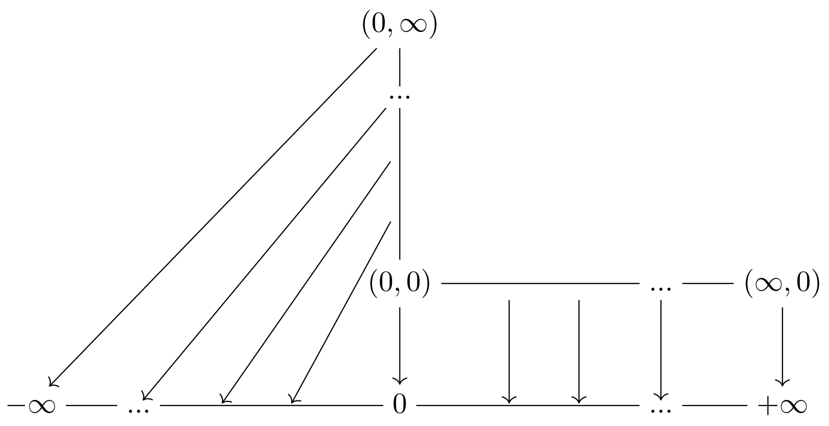
In STL we work in values $\in \overline{\mathbb{R}}$ such that gradient–based optimization treats truth values just like network activations. In order to achieve this, map the just defined possibility scores

$t \in [-1,1]$ to values $\phi(t) \in \overline{\mathbb{R}}$ with any smooth, strictly monotone, *odd* diffeomorphism $\phi : (-1,1) \longrightarrow \mathbb{R}$ satisfying:

$$\begin{aligned}
&\phi(0) = 0, && \phi(-x) = -\phi(x),\\
&\phi(x) > 0 \text{ for } x > 0, && \phi(x) > \phi(y) \text{ for } x > y,\\
&\lim_{x\to -1^+} \phi(x) = -\infty, && \lim_{x\to 1^-} \phi(x) = \infty.
\end{aligned}$$

Table 4 lists several classical choices, tanh is also used in [9] [p. 8]. The choice of ϕ depends on the application and the desired properties of the resulting logic.

Intuition We basically do exactly the same projections as before, only that everything is drawn out from the beginng by the smooth diffeomorphism ϕ to the extended real line $\overline{\mathbb{R}}$. That means our triangle Δ_1 is drawn out in two-dimensional space to an infinite one, then we project to an infinite L of axes, and then we project to the extended real line $\overline{\mathbb{R}}$ as in:



Definition 13 (STL$_\phi$ Monad $\tilde{\Pi}_\phi$) *$\tilde{\Pi}_\phi$ is defined for sets X as, with $\infty \cdot 0 := 0$:*

$$\tilde{\Pi}_\phi(X) := \big\{\tilde{\pi}(\phi \circ p) \,|\, p \in \mathcal{S}(X) : |A_p| = 1\big\} \cup \big\{0\big\} \subseteq \mathcal{S}X,$$

$$\eta_X(x) := \delta_x, \qquad k^*(\tilde{\pi}(\phi \circ p)) := \tilde{\pi}\big(\max(\phi \circ p)k(A_p)\big).$$

D.1.4. Expanded NeSy shifts diagram

Having properly defined the Possibility and STL$_\phi$ monads, we can now add them to our NeSy shifts diagram again. Note that the descriptions in parentheses are only for the case of X being finite and π means the projection as described in D.1.2:

$$\begin{tikzcd}
 & {[-1,1]}\ \text{(Possibil.)} \arrow[rr, hook, two heads, "\phi"] \arrow[dl, two heads, "-1\mapsto 0"'] & & {[-\infty,\infty]}\ \mathrm{STL}_{\phi} \\
\{0,1/2,1\}\ \text{(3-Valued)} \arrow[dr, hook] & & {[0,1]}\ \text{(Distrib.)} \arrow[ul, hook, "0\mapsto -1"'] \arrow[ll, two heads, "\mathrm{pro}"] \arrow[dl, hook, two heads, "id"'] & \\
 & {[0,1]}\ \text{(Probab.)} \arrow[rr, hook, two heads, "id"] & & {[0,1]}\ \text{(LTN)} \\
\{0,1\}\ \text{(Classical)} \arrow[uu, hook] \arrow[ur, hook] \arrow[rr, hook] & & {[0,1]}\ \text{(Fuzzy)} \arrow[ul, hook, two heads, "id"] \arrow[uu, hook, two heads, "id"'] &
\end{tikzcd}$$

$$\begin{tikzcd}
 & \tilde{\Pi}(X) \arrow[rr, hook, two heads, "\phi"] \arrow[dl, two heads, "\arg\max"'] & & \tilde{\Pi}_{\phi}(X) \\
\mathcal{P}_{\neq\emptyset}(X) \arrow[dr, hook, "(\mathrm{unif})"] & & \mathcal{D}(X) \arrow[ul, "\varphi\circ\pi"'] \arrow[ll, two heads, "\arg\max"] \arrow[dl, hook] & \\
 & \mathcal{G}(X) \arrow[rr, hook, two heads, "(id)"] & & \mathcal{O}(X) \\
\mathrm{Id}_{\mathbf{Set}}(X) \arrow[uu, hook, "\{\cdot\}"] \arrow[ur, hook, "\delta"] \arrow[rr, hook, two heads, "id"] & & \mathrm{Id}_{\mathbf{Set}}(X) \arrow[ul, hook, "\delta"] \arrow[uu, hook, "\delta"'] &
\end{tikzcd}$$

Appendix E. Fuzzy Semantics with Computational Functions

In the original ULLER paper, the fuzzy semantics of computational function symbols were introduced to relate closely to fuzzy and differentiable logics and to support the interpretation of fuzzy or differentiable predicates. In our setting those predicates are already interpreted directly in a fuzzy or differentiable manner, so this extra layer is unnecessary. Moreover, the combined use of a t-norm and a t-conorm is not actually exploited in that correspondence and therefore lacks a clear motivation. We instead propose an alternative, monadic definition that also works in the continuous setting, which was previously omitted. The details remain to be elaborated; what follows is a sketch of the underlying idea.

We will define two new monads in addition to the already mentioned identity monad for the Classical Fuzzy semantics given in Table 1.

E.1. Pseudo Giry monad

Definition 14 ($(\hat{+},\hat{\cdot})$-Giry monad) *The* $(\hat{+},\hat{\cdot})$-Giry monad $\mathcal{G}_{(\hat{+},\hat{\cdot})}$ *or pseudo-Giry monad with respect to the pseudo-multiplication* $\hat{\cdot}$ *and pseudo-addition* $\hat{+}$ *is defined to be a monad on*

Table 5: Fuzzy NeSy Frameworks

Logic/Theory	$\mathcal{C}$	$\mathcal{T}$	Ω	$\mathcal{T}\Omega$	**Alg. Struct.**
Classical Fuzzy	**Set**	Identity	$[0,1]$	$[0,1]$	BL–Alg.
Discrete $\oplus$-Fuzzy	**Set**	$\oplus$-Distribution $\mathcal{D}_\oplus$	$\{0,1\}$	$\oplus_1$	BL–Alg.
Continuous $\oplus$-Fuzzy	**QBS**	$\oplus$-Giry $\mathcal{G}_\oplus$	$\{0,1\}$	$\oplus_1$	BL–Alg.

the category of quasi-borel spaces **QBS**. *And yet, we only will talk about its restriction on the full subcategory*[13] **SMeas** *of standard measurable spaces to present it more understandably. The pseudo-Giry monad is defined as follows:*

$$\mathcal{G}_{(\hat{+},\hat{\cdot})}(X) := \{\rho : X \to [0,1] \mid \rho \textit{ is } \hat{+}\textit{-measurable}, \int_X^{\hat{+},\hat{\cdot}} \rho(x)\, dx = 1\}$$
$$\mathcal{G}_{(\hat{+},\hat{\cdot})}(f) := \big(\rho \mapsto \rho \circ f\big)$$

Instead of taking of a t-norm and t-conorm pair we have taken a pair of pseudo-addition $\hat{+}$ and pseudo-multiplication $\hat{\cdot}$ as introduced in [10]. Now we can replace the integral of the probability monad with the universal integral with respect to $\hat{+}$ and $\hat{\cdot}$, which we introduce as:

$$[\![x := m(\boldsymbol{T})(F)]\!] \;=\; \int_{a\in\mathcal{I}(s_m)}^{\hat{+},\hat{\cdot}} [\![F]\!]_{\nu[x\mapsto a]}\, d\rho_m(\,\cdot \mid \boldsymbol{T})(a). \tag{8}$$

Now, if we want to connect this to a *continuous* t-norm, t-conorm pair $(\otimes, \oplus)$ we can define $\hat{+} := \oplus$ and $\hat{\cdot}$ as any pseudo-multiplication compatible with $\oplus$. For example just take the normal (bounded) addition and normal multiplication and obtain simply the probabilistic semantics equation 1. The reason we can't generally choose $\hat{\cdot} = \otimes$ is, that most often $\otimes$ and $\oplus$ don't distribute and therefore are not compatible. If we choose a continuous t-conorm $\oplus$ and implicitly take it's dual t-norm $\otimes$ and a pseudo-multiplication $\hat{\cdot}$ compatible with $\oplus$, we name that $\mathcal{G}_\oplus$ and $\oplus$-fuzzy semantics. We also define $\oplus_1 := \{(a,b) \mid a \oplus b = 1\}$ and observe that $\mathcal{G}_\oplus(\{0,1\}) = \oplus_1$. Leaving things implicit is canonical in some cases:

E.2. Canonical pseudo–multiplications for strict continuous t-conorms

Let $S\colon [0,1]^2 \to [0,1]$ be a *strict*[14] continuous t-conorm. Then S admits an *additive generator*

$$\phi\colon [0,1] \longrightarrow [0,\infty], \qquad \phi \text{ strictly increasing, } \phi(0) = 0, \quad S(a,b) = \phi^{-1}\big(\phi(a) + \phi(b)\big).$$

Definition 15 (canonical pseudo–product) *The* pseudo–multiplication *("S-product") associated with S is the binary operation*

$$a \mathbin{\hat{\cdot}} x \;:=\; \phi^{-1}\big(\phi(a)\, x\big), \qquad a, x \in [0,1].$$

13. See [5] for a proof that this forms indeed a full subcategory.
14. Strict (a.k.a. Archimedean) means $x < y \Rightarrow S(x,t) < S(y,t)$ for every fixed $t \in (0,1]$. All continuous t-conorms used in practice besides max and the nilpotent maximum are strict.

Proposition 16 *The operation $\hat{\cdot}$ together with S satisfies the axioms* (M1)–(M5) *of [10]; hence $(S, \hat{\cdot})$ forms a* pseudo-addition / pseudo-multiplication pair*. In particular*

$$a \mathbin{\hat{\cdot}} (x \mathbin{\hat{+}} y) = (a \mathbin{\hat{\cdot}} x) \mathbin{\hat{+}} (a \mathbin{\hat{\cdot}} y), \qquad e := 1 \text{ is the left unit of } \hat{\cdot}.$$

Remark 17 *If the continuous t-conorm S is* not *strict (e.g. $S(a,b) = \max\{a,b\}$), no additive generator exists; consequently there is no unique pseudo-multiplication satisfying* (M1)–(M5)*. One must either relax the axioms or choose an operation ad hoc.*

E.3. Relation to the Finite Fuzzy Semantics of ULLER

If we now take one of these strict continuous t-conorms $\oplus$, its dual t-norm $\otimes$ and the canonical pseudo-multiplication $\widehat{\otimes}$ associated with $\oplus$, equation 8 reduces to the following formula for the interpretation of a computational function symbol $m \in \text{MFunc}_\Sigma$:

$$[\![x := m(\boldsymbol{T})(F)]\!] = \bigoplus_{a \in \mathcal{I}(s_m)} [\![F]\!]_{\nu[x \mapsto a]} \mathbin{\widehat{\otimes}} \rho_m(a \mid \boldsymbol{T}). \tag{9}$$

This closely resembles the finite fuzzy semantics of ULLER, the difference being that we use the canonical pseudo-multiplication $\widehat{\otimes}$ instead of the t-norm $\otimes$, such that this t-conorm $\oplus$ sum behaves as a monad, distributes as we would expect it to and is also just a special case of the continuous case. In analogy to $\mathcal{G}_\oplus$ the monad of use here is $\mathcal{D}_\oplus$, which is simply $\mathcal{G}_\oplus$ only with finite supported probability $\oplus$-measures, the same difference as between the Giry monad $\mathcal{G}$ and the distribution monad $\mathcal{D}$.

Table 6: Examples of pseudo–products

t-conorm $S(a,b)$	generator ϕ	pseudo–product $a \hat{\cdot} x$
Probabilistic $a+b-ab$	$-\ln(1-t)$	$1-(1-x)^{a}$
Łukasiewicz $\min\{1, a+b\}$	t	$\max\{0,\, x+a-1\}$
Einstein $\dfrac{a+b}{1+ab}$	$\frac{t}{1-t}$	$\dfrac{ax}{1+ax}$

The table 6 shows some examples of pseudo-multiplications for strict continuous t-conorms of which we already have used the probabilistic one in the probabilistic semantics in section 4.2 to define our conjunction.

Now, having defined the pseudo-Giry monads, we can also add it to our NeSy shifts diagram of section 4.3, which we had already extended in section D.1.2 and now extend it one last time. The arrows between the Giry monad $\mathcal{G}$ and the pseudo-Giry monad $\mathcal{G}_\oplus$ arises only when taking the continuous t-conorm $\oplus$ as the normal (bounded) addition, it's dual t-norm $\otimes$ as the Łukasiewicz t-norm and the pseudo-multiplication $\hat{\cdot}$ as normal multiplication, since then we simply obtain the probabilistic semantics of section 4.2 and the normal Giry monad $\mathcal{G}$:

$$\begin{tikzcd}
 & \tilde{\Pi}(X) \arrow[rr, hook, two heads, "\phi"] \arrow[dl, two heads, "\arg\max"'] & & \tilde{\Pi}_\phi(X) \\
\mathcal{P}_{\neq\emptyset}(X) \arrow[dr, hook, "\text{(unif)}"] & & \mathcal{D}(X) \arrow[ul, "\varphi\circ\pi"'] \arrow[ll, two heads, "\arg\max"'] \arrow[dl, hook] & \\
 & \mathcal{G}(X) \arrow[rr, hook, two heads, "(id)"] \arrow[dd, hook, two heads, "(\oplus=+,\otimes=Luk,\widehat{\otimes}=\cdot)"] & & \mathcal{O}(X) \\
\mathrm{Id}_{\mathbf{Set}}(X) \arrow[uu, hook, "\{\cdot\}"] \arrow[ur, hook, "\delta"] \arrow[rr, hook, two heads, "id"] \arrow[dr, hook, "\delta"] & & \mathrm{Id}_{\mathbf{Set}}(X) \arrow[uu, hook, "\delta"] \arrow[ul, hook, "\delta"'] \arrow[dl, hook, "\delta"'] \arrow[d, hook, "\delta"] & \\
 & \mathcal{G}_\oplus(X) & \mathcal{D}_\oplus(X) \arrow[l, hook] &
\end{tikzcd}$$

Appendix F. Type Checking

We explain why the formulas in Table 2 (categorical semantics) are well-typed. For a given $\mathcal{I}(f) \circ < [\![T_1]\!] \circ \pi_1, \ldots, [\![T_n]\!] \circ \pi_n >$, we define $\pi_i := \mathcal{V}_H \to \mathcal{V}_{T_i}$ as the projection on the i–th component of the product and where $H := f(T_1, \ldots, T_n)$. If for example T_1 is a constant, we have $\mathcal{V}_{T_1} = 1$ since a constant has empty context. In this case π_1 is the unique terminal arrow. Similarly, for a given pair of formulas $< [\![F]\!] \circ \pi_F, [\![G]\!] \circ \pi_G >$ we define $\pi_F : \mathcal{V}_F \times \mathcal{V}_G \to \mathcal{V}_F$ as the projection on the F–th component of the product, and for G alike. For the quantifiers we type check as follows:

$$\mathcal{V}_F = \prod_{y:t \in \Gamma_F} \mathcal{I}(t) \cong \Big(\prod_{y:t \in \Gamma_F \setminus x:s} \mathcal{I}(t) \Big) \times \mathcal{I}(s).$$

$$[\![F]\!] : V_{F \setminus x:s} \times \mathcal{I}(s) \longrightarrow \mathcal{T}\Omega, \qquad \Lambda_s\big([\![F]\!]\big) : V_{F \setminus x:s} \longrightarrow \mathcal{T}\Omega^{\mathcal{I}(s)},$$
$$\bigwedge_{\mathcal{I}(s)}, \bigvee_{\mathcal{I}(s)} : \mathcal{T}\Omega^{\mathcal{I}(s)} \longrightarrow \mathcal{T}\Omega.$$

Of course equation 1 also needs some more explanation. The $(-)^*$ operation is the lifting of the monad $\mathcal{T}$. We define for a computational function symbol $m \in \mathrm{MFunc}_\Sigma$:

$$V_{F \setminus x:m} := \prod_{y:t \in \Gamma_F \setminus x:s_m} \mathcal{I}(t).$$

where s_m is the sort of m. Set $H := (x := m(T_1, \ldots, T_n)(F))$. We also observe the following typing facts:

$$[\![F]\!] : V_{F \setminus x:m} \times \mathcal{I}(s_m) \longrightarrow \mathcal{T}\Omega, \qquad [\![F]\!]^* : \mathcal{T}\big(V_{F \setminus x:m} \times \mathcal{I}(s_m)\big) \longrightarrow \mathcal{T}\Omega.$$

$$\mathcal{S} : V_{F \setminus x:m} \times \mathcal{T}\mathcal{I}(s_m) \longrightarrow \mathcal{T}\big(V_{F \setminus x:m} \times \mathcal{I}(s_m)\big), \qquad [\![F]\!]^* \circ \mathcal{S} : V_{F \setminus x:m} \times \mathcal{T}\mathcal{I}(s_m) \longrightarrow \mathcal{T}\Omega,$$

$$\Lambda_m\big(\llbracket F\rrbracket^* \circ\ \mathcal{S}\big) : V_{F\setminus x:m}\ \longrightarrow\ \mathcal{T}\Omega^{\mathcal{T}\mathcal{I}(s_m)}, \qquad \Lambda_m\big(\llbracket F\rrbracket^* \circ\ \mathcal{S}\big)\ \circ\ \pi_{V_{F\setminus x:m}} : V_H\ \longrightarrow\ \mathcal{T}\Omega^{\mathcal{T}\mathcal{I}(s_m)}.$$

$$\mathcal{I}(m) : \prod_{i=1}^{n}\mathcal{I}(s_{T_i})\ \longrightarrow\ \mathcal{T}\big(\mathcal{I}(s_m)\big), \qquad \big\langle \llbracket T_1\rrbracket\circ\pi_1,\ \ldots,\ \llbracket T_n\rrbracket\circ\pi_n\big\rangle : V_H\ \longrightarrow\ \prod_{i=1}^{n}\mathcal{I}(s_{T_i}),$$

$$[m(T_1,\ldots,T_n)] := \mathcal{I}(m)\circ\big\langle\llbracket T_1\rrbracket\circ\pi_1,\ldots,\llbracket T_n\rrbracket\circ\pi_n\big\rangle : V_H\ \longrightarrow\ \mathcal{T}\mathcal{I}(s_m).$$

$$\mathrm{ev} : \mathcal{T}\mathcal{I}(s_m)\times\mathcal{T}\Omega^{\mathcal{T}\mathcal{I}(s_m)}\to\mathcal{T}\Omega,$$

$$\big\langle[m(T_1,\ldots,T_n)],\ \Lambda_m\big(\llbracket F\rrbracket^*\circ\mathcal{S}\big)\circ\pi_{V_{F\setminus x:m}}\big\rangle : \mathcal{V}_H\ \longrightarrow\ \mathcal{T}\mathcal{I}(s_m)\times\mathcal{T}\Omega^{\mathcal{T}\mathcal{I}(s_m)},$$

$$\mathrm{ev}\circ\big\langle[m(T_1,\ldots,T_n)],\ \Lambda_m\big(\llbracket F\rrbracket^*\circ\mathcal{S}\big)\circ\pi_{V_{F\setminus x:m}}\big\rangle : \mathcal{V}_H\ \longrightarrow\ \mathcal{T}\Omega.$$

This we can also write more compactly as:

$$\llbracket F\rrbracket^*\circ\mathcal{S}\circ\langle\pi_{V_{F\setminus x:m}},\ [m(T_1,\ldots,T_n)]\rangle : \mathcal{V}_H\ \longrightarrow\ \mathcal{T}\Omega.$$