Commuting Rules for the Later Modality and Quantifiers in Step-Indexed Logics

Bálint Kocsis Robbert Krebbers

Radboud University

TYPES, June 11, 2025

() < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < () < ()

Summary

Provide novel rules for commuting the later modality with quantifiers in step-indexed logics, sound with respect to their semantics in the topos of trees

Summary

Provide novel rules for commuting the later modality with quantifiers in step-indexed logics, sound with respect to their semantics in the topos of trees

$$\frac{\Gamma \vdash Q : \blacktriangleright (A \to \operatorname{Prop})}{\Gamma \mid \operatorname{lift}(\operatorname{next}\operatorname{ex} \circledast Q) \dashv \vdash \exists y : \blacktriangleright A. \operatorname{lift}(Q \circledast y)}$$

$$\frac{\Gamma \vdash Q : \blacktriangleright (A \to \operatorname{Prop})}{\Gamma \mid \operatorname{lift}(\operatorname{next}\operatorname{all} \circledast Q) \dashv \vdash \forall y : \blacktriangleright A. \operatorname{lift}(Q \circledast y)}$$

Summary

Provide novel rules for commuting the later modality with quantifiers in step-indexed logics, sound with respect to their semantics in the topos of trees

$$\begin{array}{c} {\displaystyle \Gamma \vdash Q : \blacktriangleright (A \to \operatorname{Prop}) \\ \\ \overline{\Gamma \mid \operatorname{lift} (\operatorname{next} \operatorname{ex} \circledast Q) \dashv} \exists y : \blacktriangleright A. \operatorname{lift} (Q \circledast y) \\ \\ \\ {\displaystyle \Gamma \vdash Q : \blacktriangleright (A \to \operatorname{Prop}) \\ \\ \overline{\Gamma \mid \operatorname{lift} (\operatorname{next} \operatorname{all} \circledast Q) \dashv} \forall y : \blacktriangleright A. \operatorname{lift} (Q \circledast y) \end{array}$$

Also, everything is formalised in Rocq ;)

Outline

1 Context

2 Guarded recursion

3 Step-indexed logic

4 Semantics

(5) The ▷ modality and quantifiers

6 Conclusion

イロト イポト イヨト イヨト

How do we ensure that a recursive definition is well-defined?

医尿道氏试道

How do we ensure that a recursive definition is well-defined?

Example

For $n \in \mathbb{N}$: $n! = 1 \cdot 2 \cdot \ldots \cdot n$.

• • • • • • • •

How do we ensure that a recursive definition is well-defined?

Example

For $n \in \mathbb{N}$: $n! = 1 \cdot 2 \cdot \ldots \cdot n$. Recursively:

$$n! = \begin{cases} 1 & \text{if } n = 0\\ n \cdot (n-1)! & \text{if } n > 0 \end{cases}$$

• • = • • = •

How do we ensure that a recursive definition is well-defined?

Example

For $n \in \mathbb{N}$: $n! = 1 \cdot 2 \cdot \ldots \cdot n$. Recursively:

$$n! = egin{cases} 1 & ext{if } n = 0 \ n \cdot (n-1)! & ext{if } n > 0 \end{cases}$$

Example

For $a \in \mathbb{N}^{\mathbb{N}}$: $f((a_n)_{n \in \mathbb{N}}) = (a_n + 1)_{n \in \mathbb{N}}$.

Kocsis, Krebbers (Radboud University)

How do we ensure that a recursive definition is well-defined?

Example

For $n \in \mathbb{N}$: $n! = 1 \cdot 2 \cdot \ldots \cdot n$. Recursively:

$$n! = egin{cases} 1 & ext{if } n = 0 \ n \cdot (n-1)! & ext{if } n > 0 \end{cases}$$

Example

For $a \in \mathbb{N}^{\mathbb{N}}$: $f((a_n)_{n \in \mathbb{N}}) = (a_n + 1)_{n \in \mathbb{N}}$. Recursively:

$$f(a) = (a_0 + 1, f(a \circ s))$$
 (where $s(n) = n + 1$)



Structural recursion on inductive data types



Structural recursion on inductive data types

Example
data N : Set where
Z : N
$S : N \rightarrow N$
$_*$: N \rightarrow N \rightarrow N
•••
fact : N -> N
fact 0 = 1
fact (S n) = fact n * S n



Structural recursion on inductive data types

Encodes well-founded definitions

Kocsis, Krebbers (Radboud University)

• • = • • = •

< □ > < 円



Structural corecursion on coinductive data types

• • = • • = •

< □ > < 円



Structural corecursion on coinductive data types

Example
<pre>record Str : Set where coinductive field hd : N tl : Str</pre>
open S
<pre>inc : Str -> Str inc a .hd = S (a .hd) inc a .tl = inc (a .tl)</pre>

• • = • • = •



Structural corecursion on coinductive data types

Example
<pre>record Str : Set where coinductive field hd : N tl : Str</pre>
open S
inc : $Str \rightarrow Str$ inc a $hd = S(a, hd)$
inc a .tl = inc (a .tl)

Encodes productive definitions

Kocsis, Krebbers (Radboud University)

• • = • • = •



• Inductive and coinductive types have to be **strictly positive** in order to guarantee termination (hence well-definedness)

Problem

- Inductive and coinductive types have to be **strictly positive** in order to guarantee termination (hence well-definedness)
- What about more exotic domains? E.g.

```
data Exp =
    Var String
    App Exp Exp
    Lam (Exp -> Exp)
```

• • = • • = •

• Key idea: step-wise approximation

- Key idea: step-wise approximation
- Step-indexing: semantic tool for stratifying recursive definitions [AM01, Ahm04]

- Key idea: step-wise approximation
- Step-indexing: semantic tool for stratifying recursive definitions [AM01, Ahm04]
- Approximation modality: modal framework for expressing self-referential formulas [Nak00]

- Key idea: step-wise approximation
- Step-indexing: semantic tool for stratifying recursive definitions [AM01, Ahm04]
- Approximation modality: modal framework for expressing self-referential formulas [Nak00]
- Marry the two [AMRV07] and coin the term "later modality"

Guarded type theory

- New type former ► (pronounced "later")
 - > Allows us to talk about data we will only have access to in the next computation step
 - Guards recursive occurrences in recursively defined types and terms

Guarded type theory

- New type former ► (pronounced "later")
 - Allows us to talk about data we will only have access to in the next computation step
 - Guards recursive occurrences in recursively defined types and terms
- Applicative structure
 - next : $A \rightarrow \blacktriangleright A$: shifts data into the future
 - ▶ $\circledast : \blacktriangleright (A \rightarrow B) \rightarrow \blacktriangleright A \rightarrow \blacktriangleright B$: applies a function in the future

• • = • • = •

Guarded type theory

- New type former ► (pronounced "later")
 - Allows us to talk about data we will only have access to in the next computation step
 - Guards recursive occurrences in recursively defined types and terms
- Applicative structure
 - ▶ next : $A \rightarrow \blacktriangleright A$: shifts data into the future
 - ▶ $\circledast : \blacktriangleright (A \to B) \to \blacktriangleright A \to \blacktriangleright B$: applies a function in the future
- Guarded fixed point combinator: fix : $(\blacktriangleright A o A) o A$
 - Self-reference delayed in time by next:

$$\texttt{fix}\,f=f\,(\texttt{next}\,(\texttt{fix}\,f))$$

• We write $\mu x : \blacktriangleright A. t$ for fix $(\lambda x : \blacktriangleright A. t)$

(4) E > (4) E >

Motivating example: streams

• $Str = \mu X . \mathbb{N} \times \blacktriangleright X$, hence $Str = \mathbb{N} \times \blacktriangleright Str$

Motivating example: streams

- $Str = \mu X . \mathbb{N} \times \blacktriangleright X$, hence $Str = \mathbb{N} \times \blacktriangleright Str$
- Constructors and destructors:

$$egin{aligned} -::-:\mathbb{N} o lackstriangle \operatorname{Str} o \operatorname{Str} \ & o \operatorname{Mt}:\operatorname{Str} o \mathbb{N} \ & o \operatorname{tl}:\operatorname{Str} o lackstriangle \operatorname{Str} \end{aligned}$$

Motivating example: streams

- $Str = \mu X . \mathbb{N} \times \blacktriangleright X$, hence $Str = \mathbb{N} \times \blacktriangleright Str$
- Constructors and destructors:

$$egin{aligned} -::-:\mathbb{N} o lackstriangle \operatorname{Str} o \operatorname{Str} \ & o \operatorname{Mt}:\operatorname{Str} o \mathbb{N} \ & o \operatorname{tl}:\operatorname{Str} o lackstriangle \operatorname{Str} \end{aligned}$$

• Recursive operations:

$$ext{zeros} = \mu s : \blacktriangleright ext{Str.0} :: s$$

 $ext{inc} = \mu r : \blacktriangleright (ext{Str} o ext{Str}). \lambda s : ext{Str.} (ext{hd} s + 1) :: (r \circledast ext{tl} s)$

• • = • • = •

э

10 / 23

Under the Curry-Howard correspondence, guarded recursive operations correspond to logical connectives and rules

Under the Curry-Howard correspondence, guarded recursive operations correspond to logical connectives and rules

- • type former $\Rightarrow \triangleright$ modality (also pronounced "later")
 - $\triangleright P$ holds now if and only if P holds at the next step

• • = • • = •

Under the Curry-Howard correspondence, guarded recursive operations correspond to logical connectives and rules

- • type former $\Rightarrow \triangleright$ modality (also pronounced "later")
 - $\triangleright \triangleright P$ holds now if and only if P holds at the next step
- Applicative structure \Rightarrow modal axioms:

$$P \vdash \triangleright P$$
 $\triangleright (P \supset Q) \vdash \triangleright P \supset \triangleright Q$

Under the Curry-Howard correspondence, guarded recursive operations correspond to logical connectives and rules

- • type former $\Rightarrow \triangleright$ modality (also pronounced "later")
 - $\triangleright \triangleright P$ holds now if and only if P holds at the next step
- Applicative structure \Rightarrow modal axioms:

$$P \vdash \rhd P$$
 $\rhd (P \supset Q) \vdash \rhd P \supset \rhd Q$

• Fixed point combinator \Rightarrow Löb rule:

$$\triangleright P \supset P \vdash P$$

In short: to prove P, we can assume that P already holds after one computation step

Crucial rules

$$P \vdash \rhd P$$
 $rac{P \vdash Q}{\rhd P \vdash \rhd Q}$ $\rhd P \supset P \vdash P$
 $\rhd (P * Q) \dashv \vdash \rhd P * \rhd Q$ $(* \in \{\land, \lor, \supset\})$ $\rhd (t =_A u) \dashv \vdash \operatorname{next} t =_{\blacktriangleright A} \operatorname{next} u$

3

Semantics

- Intuitively: sequences of approximations
 - ▶ The *n*-th element describes what the object looks like if one has only *n* steps to reason about it
 - ▶ *n*: step-index
 - \blacktriangleright and \triangleright shift step-indices

< □ > < 円

Semantics

- Intuitively: sequences of approximations
 - ▶ The *n*-th element describes what the object looks like if one has only *n* steps to reason about it
 - n: step-index
 - ► and ▷ shift step-indices
- Two main formalisms:
 - Ordered families of equivalences (used by Iris [JKJ⁺18])
 - Topos of trees

Semantics

- Intuitively: sequences of approximations
 - ▶ The *n*-th element describes what the object looks like if one has only *n* steps to reason about it
 - n: step-index
 - ► and ▷ shift step-indices
- Two main formalisms:
 - Ordered families of equivalences (used by Iris [JKJ⁺18])
 - Topos of trees
- General models of guarded recursion [BMSS12], in particular sheaves over certain complete Heyting-algebras

Topos of trees

• \mathcal{S} : presheaves on the ordinal ω

Topos of trees

- $\mathcal{S}:$ presheaves on the ordinal ω
- Objects X:

$$X_0 \xleftarrow{r_0^X} X_1 \xleftarrow{r_1^X} X_2 \xleftarrow{r_2^X} \cdots$$

Topos of trees

- $\bullet~\mathcal{S}:$ presheaves on the ordinal ω
- Objects X:

$$X_0 \xleftarrow{r_0^X} X_1 \xleftarrow{r_1^X} X_2 \xleftarrow{r_2^X}$$
Notation: $x|_n = (r_n^X \circ r_{n+1}^X \circ \ldots \circ r_{m-1}^X)(x)$
• Morphisms $f : X \to Y$:



. . .

3

Guarded recursion

• • : $S \to S$ sends X to

$$\{*\} \xleftarrow{!} X_0 \xleftarrow{r_0} X_1 \xleftarrow{r_1} \cdots$$

3

Guarded recursion

• $\blacktriangleright : S \to S$ sends X to

$$\{*\} \xleftarrow{!} X_0 \xleftarrow{r_0} X_1 \xleftarrow{r_1} \cdots$$

• next_X : X \rightarrow \next_X, (next_X)_n = $r_n^{
arrow X}$

$$X_0 \xleftarrow{r_0^X} X_1 \xleftarrow{r_1^X} X_2 \xleftarrow{r_2^X} \cdots$$

$$\downarrow \qquad r_0^X \downarrow \qquad r_1^X \downarrow$$

$$\{*\} \xleftarrow[]{} Y_0 \xleftarrow[]{} Y_0 \xleftarrow[]{} Y_1 \xleftarrow[]{} Y_1 \xleftarrow[]{} Y_1 \xleftarrow[]{} Y_1$$

Kocsis, Krebbers (Radboud University)

✓ □ > < □ > < □ > < ≥ > < ≥ >
TYPES, June 11, 2025

3

Example

• Streams:

$\mathbb{N} \xleftarrow{\pi_1} \mathbb{N} \times \mathbb{N} \xleftarrow{\pi_1} \mathbb{N} \times \mathbb{N} \times \mathbb{N} \xleftarrow{\pi_1} \cdots$

Example

• Streams:

$$\mathbb{N} \xleftarrow{\pi_1} \mathbb{N} \times \mathbb{N} \xleftarrow{\pi_1} \mathbb{N} \times \mathbb{N} \times \mathbb{N} \xleftarrow{\pi_1} \cdots$$

• hd:
$$\mathtt{Str} o \mathbb{N}$$
, $\mathtt{hd}_n(s_0,\ldots,s_n) = s_0$

• inc: $\operatorname{Str} \to \operatorname{Str}$, $\operatorname{inc}_n(s_0, \ldots, s_n) = (s_0 + 1, \ldots, s_n + 1)$

• Essentially Kripke semantics over the natural numbers

- Essentially Kripke semantics over the natural numbers
- Truth of a proposition depends on the step-index n
- P holds at n if it is true for n steps

- Essentially Kripke semantics over the natural numbers
- Truth of a proposition depends on the step-index n
- P holds at n if it is true for n steps
- If P is true for n steps, then it is also true for less than n steps
- Hence: a truth value is a downward closed subset of step indices Equivalently (classically), a conatural number

4 E K 4 E K

Propositions in S

• Define the object Ω as

$$\{0,1\} \xleftarrow{r_0^\Omega} \{0,1,2\} \xleftarrow{r_1^\Omega} \{0,1,2,3\} \xleftarrow{r_2^\Omega} \cdots$$

where $r_n^{\Omega}(m) = \min(m, n+1)$. • $\triangleright: \Omega \to \Omega$ is given by $\triangleright_n(m) = \min(m+1, n+1)$

э

Propositions in S

• Define the object Ω as

$$\{0,1\} \xleftarrow{r_0^\Omega} \{0,1,2\} \xleftarrow{r_1^\Omega} \{0,1,2,3\} \xleftarrow{r_2^\Omega} \cdots$$

where $r_n^{\Omega}(m) = \min(m, n+1)$. • $\triangleright: \Omega \to \Omega$ is given by $\triangleright_n(m) = \min(m+1, n+1)$

э

Propositions in \mathcal{S}

 $\bullet\,$ Define the object Ω as

$$\{0,1\} \xleftarrow{r_0^\Omega} \{0,1,2\} \xleftarrow{r_1^\Omega} \{0,1,2,3\} \xleftarrow{r_2^\Omega} \cdots$$

where $r_n^{\Omega}(m) = \min(m, n+1)$.

- $\rhd : \Omega \to \Omega$ is given by $\rhd_n(m) = \min(m+1, n+1)$
- Forcing relation: for $P: X \to \Omega$, $n \in \mathbb{N}$, and $x \in X_n$, we define

$$n, x \Vdash P \iff n \in P_n(x)$$

7 E K 7 E K

э

Propositions in \mathcal{S}

 $\bullet\,$ Define the object Ω as

$$\{0,1\} \xleftarrow{r_0^\Omega} \{0,1,2\} \xleftarrow{r_1^\Omega} \{0,1,2,3\} \xleftarrow{r_2^\Omega} \cdots$$

where $r_n^{\Omega}(m) = \min(m, n+1)$.

- $arphi:\Omega
 ightarrow\Omega$ is given by $arphi_n(m)=\min(m+1,n+1)$
- Forcing relation: for $P: X \to \Omega$, $n \in \mathbb{N}$, and $x \in X_n$, we define

$$n, x \Vdash P \iff n \in P_n(x)$$

• Crucially: $n+1, x \Vdash P \iff n, x|_n \Vdash P$

7 E K 7 E K

18 / 23

\triangleright and quantifiers

• We have

$$\exists x : A. \rhd P \vdash \rhd (\exists x : A. P) \qquad \qquad \rhd (\forall x : A. P) \vdash \forall x : A. \rhd P$$

\triangleright and quantifiers

• We have

$$\exists x : A. \rhd P \vdash \rhd (\exists x : A. P) \qquad \qquad \rhd (\forall x : A. P) \vdash \forall x : A. \rhd P$$

• However, the other directions are not valid:

$$n+1 \Vdash \rhd (\exists x : A. P) \iff \exists a \in A_n. n, a \Vdash P$$
$$n+1 \Vdash \exists x : A. \rhd P \iff \exists a \in A_{n+1}. n, a|_n \Vdash P$$

э

19/23

イロト イポト イヨト イヨト

\triangleright and quantifiers

• We have

$$\exists x : A. \triangleright P \vdash \triangleright (\exists x : A. P) \qquad \qquad \triangleright (\forall x : A. P) \vdash \forall x : A. \triangleright P$$

• However, the other directions are not valid:

$$n+1 \Vdash \rhd (\exists x : A. P) \iff \exists a \in A_n. n, a \Vdash P$$
$$n+1 \Vdash \exists x : A. \rhd P \iff \exists a \in A_{n+1}. n, a|_n \Vdash P$$

• There does not seem to be a general rule for commuting \triangleright with a quantifier

イロト 不得下 イヨト イヨト

- Require that A be total and inhabited
 - Semantic condition: A_0 is inhabited and all r_n^A are surjective

- Require that A be total and inhabited
 - Semantic condition: A_0 is inhabited and all r_n^A are surjective
- A is total and inhabited \iff next is internally surjective:

 $\mathtt{TI}(A) := \forall y : \blacktriangleright A. \exists x : A. \mathtt{next} x = y$

- Require that A be total and inhabited
 - Semantic condition: A_0 is inhabited and all r_n^A are surjective
- A is total and inhabited \iff next is internally surjective:

 $\mathtt{TI}(A) := \forall y : \blacktriangleright A. \exists x : A. \mathtt{next} x = y$

• The following rule is valid in \mathcal{S} [BMSS12]:

$$\frac{\vdash \mathrm{TI}(A)}{\Gamma \mid \rhd (\exists x : A. P) \vdash \exists x : A. \rhd P}$$

(4) E > (4) E >

3

- Require that A be total and inhabited
 - Semantic condition: A_0 is inhabited and all r_n^A are surjective
- A is total and inhabited \iff next is internally surjective:

 $\mathtt{TI}(A) := \forall y : \blacktriangleright A. \exists x : A. \mathtt{next} x = y$

• The following rule is valid in \mathcal{S} [BMSS12]:

$$\frac{\vdash \mathrm{TI}(A) \quad \Gamma, x : A \vdash P : \mathrm{Prop}}{\Gamma \mid \rhd (\exists x : A. P) \vdash \exists x : A. \rhd P}$$

• Issue: only works for total and inhabited types

- Require that A be total and inhabited
 - Semantic condition: A_0 is inhabited and all r_n^A are surjective
- A is total and inhabited \iff next is internally surjective:

 $TI(A) := \forall y : \blacktriangleright A. \exists x : A. next x = y$

• The following rule is valid in \mathcal{S} [BMSS12]:

$$\frac{\vdash \mathrm{TI}(A) \quad \Gamma, x : A \vdash P : \mathrm{Prop}}{\Gamma \mid \rhd (\exists x : A. P) \vdash \exists x : A. \rhd P}$$

- Issue: only works for total and inhabited types
- Remark: following rule also valid in S:

$$\frac{\Gamma, x : A \vdash P : \text{Prop}}{\Gamma \mid \text{TI}(A) \land \rhd (\exists x : A. P) \vdash \exists x : A. \rhd P}$$

• We can decompose \triangleright as $\triangleright = \texttt{lift} \circ \texttt{next}$ [BMSS12, CBGB16], where

$$extsf{lift:} igstarrow \Omega o \Omega \ extsf{lift}_0(*) = 1 \ extsf{lift}_{n+1}(m) = m+1 \ extsf{lift}_{n+1}(m) = m+1$$

• • = • • = •

< □ > < 円

• We can decompose \triangleright as $\triangleright = \texttt{lift} \circ \texttt{next}$ [BMSS12, CBGB16], where

$$ext{lift:} igstarrow \Omega o \Omega \ ext{lift}_0(*) = 1 \ ext{lift}_{n+1}(m) = m+1 \ ext{}$$

• Hence, we could investigate the properties of lift

Novel commuting rules

$$\frac{\Gamma \vdash Q : \blacktriangleright (A \to \operatorname{Prop})}{\Gamma \mid \operatorname{lift}(\operatorname{next}\operatorname{ex} \circledast Q) \dashv \vdash \exists y : \blacktriangleright A. \operatorname{lift}(Q \circledast y)} \\
\frac{\Gamma \vdash Q : \blacktriangleright (A \to \operatorname{Prop})}{\Gamma \mid \operatorname{lift}(\operatorname{next}\operatorname{all} \circledast Q) \dashv \vdash \forall y : \blacktriangleright A. \operatorname{lift}(Q \circledast y)}$$

where

$$\texttt{ex} = \lambda P : A \rightarrow \texttt{Prop.} \exists x : A. P x$$
$$\texttt{all} = \lambda P : A \rightarrow \texttt{Prop.} \forall x : A. P x$$

Kocsis, Krebbers (Radboud University)

3

Novel commuting rules

$$\frac{\Gamma \vdash Q : \blacktriangleright (A \to \operatorname{Prop})}{\Gamma \mid \operatorname{lift}(\operatorname{next}\operatorname{ex} \circledast Q) \dashv \vdash \exists y : \blacktriangleright A. \operatorname{lift}(Q \circledast y)} \\
\frac{\Gamma \vdash Q : \blacktriangleright (A \to \operatorname{Prop})}{\Gamma \mid \operatorname{lift}(\operatorname{next}\operatorname{all} \circledast Q) \dashv \vdash \forall y : \blacktriangleright A. \operatorname{lift}(Q \circledast y)}$$

where

$$ex = \lambda P : A \to \text{Prop.} \exists x : A. P x$$
$$all = \lambda P : A \to \text{Prop.} \forall x : A. P x$$

- \bullet Sound in ${\cal S}$
- Imply previous rules

Kocsis, Krebbers (Radboud University)



The operation lift seems to be more fundamental and better behaved than \triangleright

< E



The operation lift seems to be more fundamental and better behaved than \vartriangleright

Future work:

• Find appropriate rules for lift

Conclusion

The operation lift seems to be more fundamental and better behaved than \vartriangleright

Future work:

- Find appropriate rules for lift
- Investigate the applicability of the rules, e.g. by formalizing a model of Iris [JKJ⁺18] in this logic

Conclusion

The operation lift seems to be more fundamental and better behaved than \vartriangleright

Future work:

- Find appropriate rules for lift
- Investigate the applicability of the rules, e.g. by formalizing a model of Iris [JKJ⁺18] in this logic
- Check if the new rules also hold in other models of step-indexing (e.g. Transfinite Iris [SGG⁺21])

The operation lift seems to be more fundamental and better behaved than \vartriangleright

Future work:

- Find appropriate rules for lift
- Investigate the applicability of the rules, e.g. by formalizing a model of Iris [JKJ⁺18] in this logic
- Check if the new rules also hold in other models of step-indexing (e.g. Transfinite Iris [SGG⁺21])
- Connect lift to $\widehat{\triangleright} : \blacktriangleright U \to U$ in guarded dependent type theory [BGC⁺16]

The operation lift seems to be more fundamental and better behaved than \vartriangleright

Future work:

- Find appropriate rules for lift
- Investigate the applicability of the rules, e.g. by formalizing a model of Iris [JKJ⁺18] in this logic
- Check if the new rules also hold in other models of step-indexing (e.g. Transfinite Iris [SGG⁺21])
- Connect lift to $\widehat{\rhd}$: $\blacktriangleright U \to U$ in guarded dependent type theory [BGC⁺16]
- Investigate analogues/generalisations in modal type theory [GKNB21]

< ∃⇒

References I

- Amal J. Ahmed. Semantics of types for mutable state. PhD thesis, Princeton University, 2004.
- Andrew W. Appel and David A. McAllester. An indexed model of recursive types for foundational proof-carrying code. ACM Trans. Program. Lang. Syst., 23(5):657-683, 2001.
- Andrew W. Appel, Paul-André Melliès, Christopher D. Richards, and Jérôme Vouillon. A very modal model of a modern, major, general type system. In POPL, pages 109–122, 2007.
- Ales Bizjak, Hans Bugge Grathwohl, Ranald Clouston, Rasmus Ejlers Møgelberg, and Lars Birkedal

Guarded dependent type theory with coinductive types. In FoSSaCS, volume 9634 of LNCS, pages 20-35, 2016.

3

References II

- Lars Birkedal, Rasmus Ejlers Møgelberg, Jan Schwinghammer, and Kristian Støvring. First steps in synthetic guarded domain theory: step-indexing in the topos of trees. *Log. Methods Comput. Sci.*, 8(4), 2012.
- Ranald Clouston, Ales Bizjak, Hans Bugge Grathwohl, and Lars Birkedal. The guarded lambda-calculus: Programming and reasoning with guarded recursion for coinductive types.

Log. Methods Comput. Sci., 12(3), 2016.

Daniel Gratzer, G. A. Kavvos, Andreas Nuyts, and Lars Birkedal. Multimodal dependent type theory. Log. Methods Comput. Sci., 17(3), 2021.

(4) E > (4) E >

References III

Ralf Jung, Robbert Krebbers, Jacques-Henri Jourdan, Ales Bizjak, Lars Birkedal, and Derek Dreyer.

Iris from the ground up: A modular foundation for higher-order concurrent separation logic. *J. Funct. Program.*, 28:e20, 2018.

Hiroshi Nakano.
 A modality for recursion.
 In *LICS*, pages 255–266, 2000.

Simon Spies, Lennard G\u00e4her, Daniel Gratzer, Joseph Tassarotti, Robbert Krebbers, Derek Dreyer, and Lars Birkedal.
 Transfinite Iris: Resolving an existential dilemma of step-indexed separation logic.
 In *PLDI*, pages 80–95, 2021.

(4) E > (4) E >