Università di Genova | DIPARTIMENTO INFORMATICA, BIOINGEGNERIA, ROBOTICA E INGEGNERIA DEI SISTEMI

Computer Science Workshop
PhD program in Computer Science and Systems Engineering

Inference Systems with Corules for Combining Safety and Liveness Properties of Session Types

Luca Ciccone, Luca Padovani

Introduction

Properties of communication protocols usually combine safety and liveness aspects. Characterizing such combined properties by means of a single inference system is difficult because of the fundamentally different techniques (coinduction and induction, respectively) usually involved in defining and proving them. Furthermore, it is not surprising that their formalization in theorem provers is a challenging task. We apply Generalized Inference Systems for defining such properties in the context of binary Session Types and we provide an Agda formalization of all the obtained results.

Goal

- Apply inference systems in the context of binary **session types** [2]
  - Properties that require induction/coinduction
- Using flexible coinduction (**generalized inference systems**) [1]
- Formalizing definitions and proofs in the proof assistant **Agda**

Flexible Coinduction

- $\langle \mathcal{I}, \mathcal{I}_{\text{co}} \rangle$ $\mathcal{I}$ set of rules, $\mathcal{I}_{\text{co}}$ set of corules

$$\frac{}{\mathit{maxElem}\ (x, x{:}\Lambda)} \qquad \frac{\mathit{maxElem}\ (x, xs)}{\mathit{maxElem}\ (z, y{:}xs)}\ z = \max(x, y) \qquad \frac{}{\mathit{maxElem}\ (x, x{:}xs)}$$

Interpretation generated by corules

$\mathsf{Gen}[\![\mathcal{I}, \mathcal{I}_{\text{co}}]\!] = \mathsf{CoInd}[\![\mathcal{I}_{|\mathsf{Ind}[\![\mathcal{I} \cup \mathcal{I}_{\text{co}}]\!]}]\!]$

- judgments with possibly infinite proof tree in $\mathcal{I}$
- where each node has a finite proof tree in $\mathsf{Ind}[\![\mathcal{I} \cup \mathcal{I}_{\text{co}}]\!]$

$\Rightarrow$ **Bounded coinduction principle** for proving the *completeness*

A Library for Flexible Coinduction in Agda

(Co)inductive predicates

built-in support, correspondence with inference systems on paper

Flexible coinduction

code duplication, no correspondence with inference systems on paper

```
record MetaRule {ℓc ℓp : Level} (U : Set ℓu) : Set _ where
  field
    Ctx : Set ℓc
    Pos : Set ℓp
    prems : Ctx → Pos → U
    conclu : Ctx → U

record IS {ℓc ℓp ℓn : Level} (U : Set ℓu) : Set _ where
  field
    Names : Set ℓn
    rules : Names → MetaRule {ℓc} {ℓp} U
```

Properties of Concurrent Systems

- Properties are usually divided in **safety** and **liveness** ones [3]

Safety

- *Something bad will never happen*
- Invariance argument - Coinductive reasoning

Liveness

- *Something good will eventually happen*
- Well-foundedness argument - Inductive reasoning

Session Types

$$T, S \ ::= \ \mathsf{nil} \mid ?f \mid !f \qquad \frac{}{?f \xrightarrow{?x} f(x)} \qquad \frac{}{!f \xrightarrow{!x} f(x)}\ x \in \mathsf{dom}(f)$$

- f is a total function (*continuation*) from a set of messages $\mathbb{V}$ to session types
- nil: unusable channel - impossible output, unexpected input (rules are not symmetric)
- $?\mathsf{end} \stackrel{\text{def}}{=} ?\emptyset.\mathsf{nil}$ different from $!\mathsf{end} \stackrel{\text{def}}{=} !\emptyset.\mathsf{nil} \Rightarrow ?\emptyset.\mathsf{nil}$ can still go to nil

Poperties of Binary Session Types

1. Weak Termination: T preserves the possibility of successfully terminating along all of its transitions
   - Safety: the set is closed by transitions
   - Liveness: each element of the set eventually reaches termination
2. Fair Compliance: T and S are fair compliant if $T \| S$ preserves the possibility of reaching a state in which T successfully terminates and S does not fail
   - Safety: the set is closed by session transitions
   - Liveness: finite interaction extension to the desired state
3. Fair Subtyping: T is a fair subtype of S if R fair compliant with T implies R fair compliant with S for every R (Liveness preserving) [4]

Fair Compliance

- Inference System $\langle \mathcal{C}, \mathcal{C}_{\text{co}} \rangle$

$$\frac{f(x) \dashv g(x)\ ^{(\forall x \in \mathsf{dom}(g))}}{?f \dashv\ !g}\ \mathsf{dom}(g) \neq \emptyset \qquad \frac{f(x) \dashv g(x)\ ^{(\forall x \in \mathsf{dom}(f))}}{!f \dashv\ ?g}\ \mathsf{dom}(f) \neq \emptyset$$

$$\frac{}{!\mathsf{end} \dashv T}\ T \neq \mathsf{nil} \qquad \frac{f(x) \dashv g(x)}{?f \dashv\ !g}\ x \in \mathsf{dom}(g) \qquad \frac{f(x) \dashv g(x)}{!f \dashv\ ?g}\ x \in \mathsf{dom}(f)$$

Proofs

- T is *compliant* with S if and only if $(T, S) \in \mathsf{CoInd}[\![\mathcal{C}]\!]$
- T is *fair compliant* with S if and only if $(T, S) \in \mathsf{Gen}[\![\mathcal{C}, \mathcal{C}_{\text{co}}]\!]$

Examples

$R = ?\mathsf{true}.R + ?\mathsf{false}.!\mathsf{end} \qquad S = !\mathsf{true}.S \qquad T = !\mathsf{true}.T \oplus !\mathsf{false}.?\mathsf{end}$

- R fair compliant with $S \Rightarrow (R, S) \in \mathsf{CoInd}[\![\mathcal{C}]\!]$ and $(R, S) \notin \mathsf{Gen}[\![\mathcal{C}, \mathcal{C}_{\text{co}}]\!]$
- R fair compliant with $T \Rightarrow (R, T) \in \mathsf{CoInd}[\![\mathcal{C}]\!]$ and $(R, T) \in \mathsf{Gen}[\![\mathcal{C}, \mathcal{C}_{\text{co}}]\!]$

Forthcoming Research

1. Type System: fair subtyping and application of corules to reason about liveness properties of (well-typed) processes (accepted at POPL 2022)
2. Generalize the results to multiparty session types

References

[1] Davide Ancona, Francesco Dagnino, and Elena Zucca. Generalizing inference systems by coaxioms. In *ESOP 2017*.

[2] Kohei Honda. Types for dyadic interaction. In *CONCUR 1993*.

[3] S. Owicki and L. Lamport. Proving liveness properties of concurrent programs. *ACM Trans. Program. Lang. Syst.*, 1982.

[4] Luca Padovani. Fair subtyping for open session types. In *ICALP 2013*.

REFERENCE ARTICLES

Luca Ciccone and Luca Padovani. Inference Systems with Corules for Fair Subtyping and Liveness Properties of Binary Session Types. ICALP21

Luca Ciccone, Francesco Dagnino and Elena Zucca. Flexible Coinduction in Agda. ITP 2021

AGDA LIBRARY AND FORMALIZATION

CONTACTS

Luca Ciccone
luca.ciccone@unito.it

Luca Padovani
luca.padovani@unito.it