Towards Behavioural Types as Coalgebras

Henning Basold — Joint work with Alex Keizer and Jorge A. Pérez

EuroProofNet WG6 Kick-Off Meeting, Stockholm

2022 May 21

Introduction

Goal

Framework for typing channel-based concurrency

- Coalgebras model communication protocols as channel behaviour
- Functor determines observable channel behaviour
- Type of a channel x used by process p is a *state* T in a coalgebra

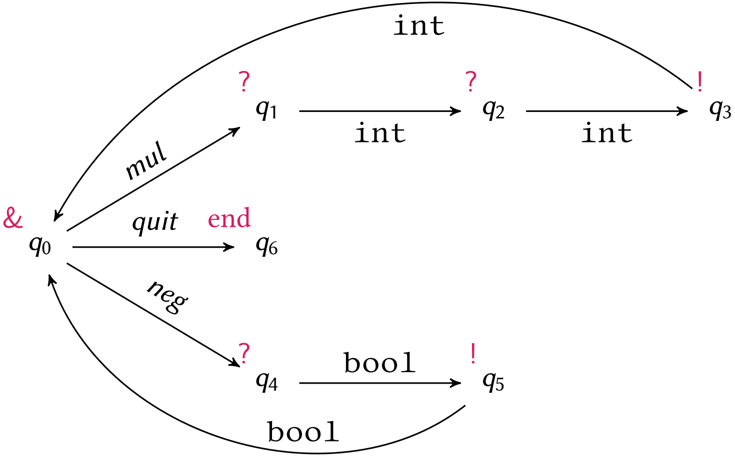
$$x : T \vdash p$$

- Coinductive predicates for subtyping, duality etc.
- Type system and syntax from finitely generated coalgebras (c.f., regular expressions)

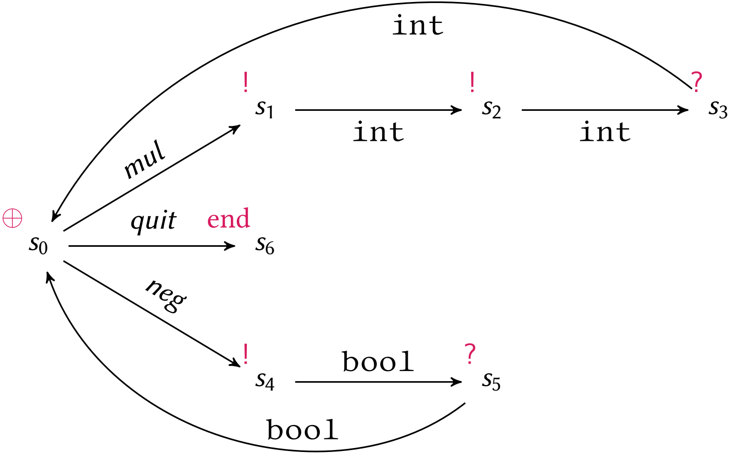
$$\begin{tikzcd}
\text{Type syntax} \arrow[rr, leftrightarrow] \arrow[dr, leftrightarrow] & & \text{Finite coalgebras} \arrow[dl, leftrightarrow] \\
& \text{Rational fixed point} &
\end{tikzcd}$$

- Semantics via coalgebraic techniques (final coalgebras, traces)

Protocol View of Mathematical Server



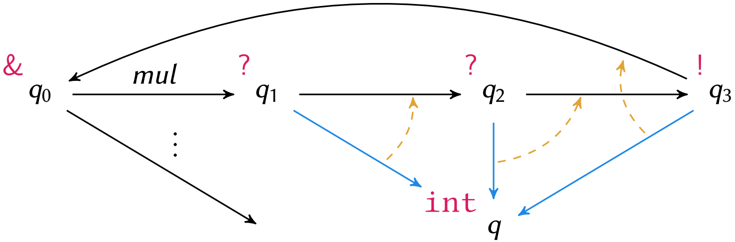
A Client Protocol for Mathematical Server



Subtyping: Another Client Protocol for Mathematical Server

$$\overset{\oplus}{r_0} \xrightarrow{\mathit{mul}} \overset{!}{r_1} \xrightarrow{\texttt{int}} \overset{!}{r_2} \xrightarrow{\texttt{int}} \overset{?}{r_3} \xrightarrow{\texttt{real}} \overset{\oplus}{r_4} \xrightarrow{\mathit{quit}} \overset{\mathrm{end}}{r_5}$$

Session delegation and session coalgebras



A Session Type System (Vasconcelos)

$$
\begin{array}{lcl}
p & ::= & ?T.\ T \\
 & | & !T.\ T \\
 & | & \&\{l_i : T_i\}_{i \in I} \\
 & | & \oplus\{l_i : T_i\}_{i \in I} \\
 & & \\
q & ::= & \mathsf{lin} \mid \mathsf{un}
\end{array}
\qquad
\begin{array}{lcl}
T & ::= & d \in D \\
 & | & \mathrm{end} \\
 & | & q\, p.T \\
 & | & X \in \mathsf{Var} \\
 & | & \mu X.T
\end{array}
$$

- D – set of basic data types (`int` etc.)
- I – finite sets of labels

Mathematical server:

$$
\mu X.\ \& \begin{cases} \mathit{mul}: & ?\mathtt{int}.\ ?\mathtt{int}.\ !\mathtt{int}.\ X \\ \mathit{neg}: & ?\mathtt{bool}.\ !\mathtt{bool}.\ X \\ \mathit{quit}: & \mathrm{end} \end{cases}
$$

Session Coalgebra Functor

- Operations: $O = \{\mathrm{com}, \mathrm{branch}, \mathrm{end}, \mathrm{bsc}, \mathrm{par}\}$
- Polarities: $P = \{\mathit{in}, \mathit{out}\}$
- Outputs A and branching B

$$
\begin{aligned}
A = \quad & \{\mathrm{com}\} \times P & \qquad B_{\mathrm{com},p} &= \{*, \mathrm{d}\} \\
& \cup \{\mathrm{branch}\} \times P \times \mathcal{P}^{+}_{\omega}(\mathbb{L}) & B_{\mathrm{branch},p,L} &= L \\
& \cup \{\mathrm{end}\} & B_{\mathrm{end}} &= \emptyset \\
& \cup \{\mathrm{bsc}\} \times D & B_{\mathrm{bsc},d} &= \emptyset \\
& \cup \{\mathrm{par}\} & B_{\mathrm{par}} &= \mathbb{1}
\end{aligned}
$$

$$
F \colon \mathbf{Set} \to \mathbf{Set} \qquad F(X) = \coprod_{a \in A} X^{B_a}
$$

Session Types form Session Coalgebra

Session Coalgebra: Coalgebra for polynomial functor

$$c \colon X \to FX$$

Syntactic Session Coalgebra

We can give a coalgebra

$$c_{\mathsf{Type}} : \mathsf{Type} \to F(\mathsf{Type})$$

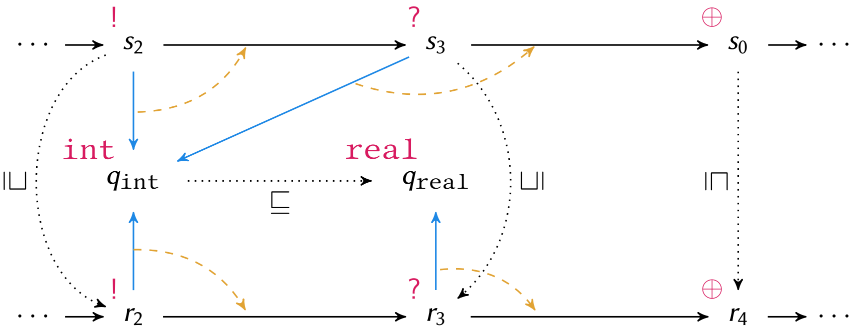
and then derive subtyping, type checking etc. as special cases for this coalgebra.

Subtyping in Session Coalgebras

Subtyping is a coinductive relation induced by $h_{\sqsubseteq} : \mathrm{Rel}_X \to \mathrm{Rel}_{F(X)}$

$$\begin{array}{rll} h_{\sqsubseteq}(R) = & \{\ ((\mathrm{com}, \mathit{in}, f), (\mathrm{com}, \mathit{in}, g)) & \mid f(*)\ R\ g(*) \text{ and } f(\mathrm{d})\ R\ g(\mathrm{d})\ \} \\ \cup & \{\ ((\mathrm{com}, \mathit{out}, f), (\mathrm{com}, \mathit{out}, g)) & \mid f(*)\ R\ g(*) \text{ and } g(\mathrm{d})\ R\ f(\mathrm{d})\ \} \\ \cup & \{\ ((\mathrm{bsc}, d, f_\emptyset), (\mathrm{bsc}, d', f_\emptyset)) & \mid d \leq_D d'\ \} \\ \cup & \{\ ((\mathrm{end}, f_\emptyset), (\mathrm{end}, f_\emptyset))\ \} & \\ & \vdots & \end{array}$$

Subtyping Example



Decidability

Definition

For any state x of a coalgebra (X, c), the *generated coalgebra* $\langle x \rangle$ is the smallest subset of X which includes x and is closed under transitions.

Definition

A coalgebra (X, c) is *finitely generated* if $\langle x \rangle$ is finite for all $x \in X$.

Lemma

The coalgebra of types $(\mathsf{Type}, c_{\mathsf{Type}})$ *is a finitely generated coalgebra.*

Theorem

Equivalence, duality and subtying are decidable for finitely generated session coalgebras.

Type System for the π-Calculus

$$\frac{c(T) = (?, f) \qquad \Gamma,\ y : U,\ x : f(*) \vdash P \qquad f(\mathrm{d}) \sqsubseteq U}{\Gamma, x : T \vdash x(y).P} \quad [\text{T-In}]$$

$$\frac{c(T) = (!, f) \qquad \Gamma,\ x : f(*) \vdash P \qquad U \sqsubseteq f(\mathrm{d})}{\Gamma, x : T, y : U \vdash \overline{x}\langle y \rangle.P} \quad [\text{T-Out}]$$

$$\frac{\Gamma_1 \vdash P \qquad \Gamma_2 \vdash Q}{\Gamma_1 \circ \Gamma_2 \vdash P \mid Q} \quad [\text{T-Rep}]$$

$\vdots$

Other Cool Things

- Definition of duality straightforward – was wrong for a while on syntax
- Syntax via rational fixed point (analogous to regular expressions), well almost
- Context-free session types (sequential composition) via coalgebras for functor G

$$C_a = \begin{cases} \{\mathrm{d}\}, & a = (\mathrm{com}, p) \\ B_a, & \text{otherwise} \end{cases} \qquad G(X) = \left(\coprod_{a \in A} X^{C_a} \right) \times X^*$$

  that adds a stack
- Functorial trace semantics and determinisation (remove stack)
- Paper: "Session Coalgebras: A Coalgebraic View on Session Types and Communication Protocols", To appear in TOPLAS or on my website

Future

- Generalise framework to other protocol presentations: typestates, mailbox types, other behavioural types, interface automata, Reo, etc.
- Example: Using presheaves instead of **Set** as base category for access permissions
- Example: Dependent types via fibrations
- Prove subject reduction etc. in general
- Investigate global properties like deadlocks
- Two levels of coalgebras:
  (1) Types/protocols of channels
  (2) Processes that use typed channels
- Coalgebraic approach to linear logic