Towards Combinatory Logic Synthesis

Jakob Rehof (Dortmund)

Based on joint work with
B. Düdder, M. Martens (Dortmund), P. Urzyczyn (Warsaw)
and with thanks to Roger Hindley

BEAT'13, January 22 2013, Rome, Italy

This Talk

- The applied research programme behind recent theoretical works on inhabitation [1] [2] [3]
- Behavioural type specification with intersection types and combinatory logic for composition synthesis

---

[1] B.Düdder, M.Martens, J.Rehof, P. Urzyczyn: *Bounded Combinatory Logic*. CSL Computer Science Logic, 2012.

[2] J.Rehof, P.Urzyczyn: *Finite Combinatory Logic with Intersection Types*. TLCA Typed Lambda Calculi and Applications, 2011.

[3] J. Rehof, P. Urzyczyn: *The Complexity of Inhabitation with Explicit Intersection*. Logic and Program Semantics. Essays Dedicated to Dexter Kozen, 2012.

Combinatory Logic Synthesis

The typability relation in combinatory logic (CL)

$$\Gamma \vdash e : \tau$$

as logical model of applicative composition e of named *component interfaces* ($\mathtt{F} : \sigma$) from a *repository* Γ satisfying a specification τ.

Combinatory Logic Synthesis

The typability relation in combinatory logic (CL)

$$\Gamma \vdash e : \tau$$

as logical model of applicative composition e of named *component interfaces* ($\mathtt{F} : \sigma$) from a *repository* Γ satisfying a specification τ.

The inhabitation relation

$$\Gamma \vdash ? : \tau$$

as logical model of *composition synthesis* goal specification: *does there exist a composition e with* $\Gamma \vdash e : \tau$?

Combinatory Logic Synthesis

The typability relation in combinatory logic (CL)

$$\Gamma \vdash e : \tau$$

as logical model of applicative composition e of named *component interfaces* ($\mathtt{F} : \sigma$) from a *repository* Γ satisfying a specification τ.

The inhabitation relation

$$\Gamma \vdash ? : \tau$$

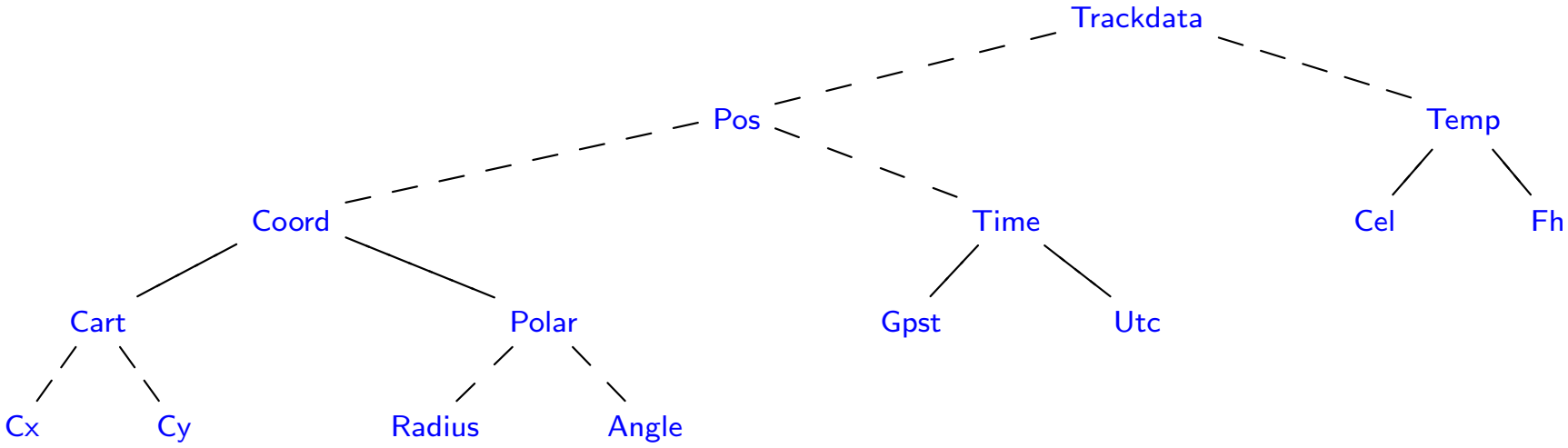
as logical model of *composition synthesis* goal specification: *does there exist a composition* e *with* $\Gamma \vdash e : \tau$?

$$\frac{}{\Gamma, X : A \vdash_s X : S(A)}(\mathsf{var}) \qquad \frac{\Gamma \vdash_s e : A \to B \quad \Gamma \vdash_s e' : A}{\Gamma \vdash_s (e\ e') : B}(\to\mathsf{E})$$

Example Repository

$$
\begin{array}{lcl}
\texttt{Tr} & : & () \to \mathtt{D}((\mathtt{R},\mathtt{R}),\mathtt{R},\mathtt{R}) \\
\texttt{pos} & : & \mathtt{D}((\mathtt{R},\mathtt{R}),\mathtt{R},\mathtt{R}) \to ((\mathtt{R},\mathtt{R}),\mathtt{R}) \\
\texttt{cdn} & : & ((\mathtt{R},\mathtt{R}),\mathtt{R}) \to (\mathtt{R},\mathtt{R}) \\
\texttt{fst} & : & (\mathtt{R},\mathtt{R}) \to \mathtt{R} \\
\texttt{snd} & : & (\mathtt{R},\mathtt{R}) \to \mathtt{R} \\
\texttt{tmp} & : & \mathtt{D}((\mathtt{R},\mathtt{R}),\mathtt{R},\mathtt{R}) \to \mathtt{R} \\
\texttt{cc2pl} & : & (\mathtt{R},\mathtt{R}) \to (\mathtt{R},\mathtt{R}) \\
\texttt{cl2fh} & : & \mathtt{R} \to \mathtt{R}
\end{array}
$$

Semantics: concepts and behaviour



Semantic repository: superposition with $\cap$

$$
\begin{array}{lcl}
\Gamma = \{ & & \\
\texttt{Tr} & : & () \to \texttt{D}((\texttt{R},\texttt{R}) \cap \mathit{Cart}, \texttt{R} \cap \mathit{Gpst}, \texttt{R} \cap \mathit{Cel}) \\
\texttt{pos} & : & \texttt{D}((\texttt{R},\texttt{R}) \cap \varepsilon, \texttt{R} \cap \eta, \texttt{R}) \to ((\texttt{R},\texttt{R}) \cap \varepsilon, \texttt{R} \cap \eta) \cap \mathit{Pos} \\
\texttt{cdn} & : & ((\texttt{R},\texttt{R}) \cap \varepsilon, \texttt{R}) \cap \mathit{Pos} \to (\texttt{R},\texttt{R}) \cap \varepsilon \\
\texttt{fst} & : & ((\texttt{R},\texttt{R}) \cap \mathit{Coord} \to \texttt{R}) \cap \\
 & & (\mathit{Cart} \to \mathit{Cx}) \cap (\mathit{Polar} \to \mathit{Radius}) \\
\texttt{snd} & : & ((\texttt{R},\texttt{R}) \cap \mathit{Coord} \to \texttt{R}) \cap \\
 & & (\mathit{Cart} \to \mathit{Cy}) \cap (\mathit{Polar} \to \mathit{Angle}) \\
\texttt{tmp} & : & \texttt{D}((\texttt{R},\texttt{R}), \texttt{R}, \texttt{R} \cap \varepsilon) \to \texttt{R} \cap \varepsilon \\
\texttt{cc2pl} & : & (\texttt{R},\texttt{R}) \cap \mathit{Cart} \to (\texttt{R},\texttt{R}) \cap \mathit{Polar} \\
\texttt{cl2fh} & : & \texttt{R} \cap \mathit{Cel} \to \texttt{R} \cap \mathit{Fh} \\
\} & &
\end{array}
$$

Synthesis via inhabitation

$$
\begin{array}{lcl}
\Gamma = \{ & & \\
\texttt{Tr} & : & () \to \mathtt{D}((\mathtt{R},\mathtt{R}) \cap \mathit{Cart}, \mathtt{R} \cap \mathit{Gpst}, \mathtt{R} \cap \mathit{Cel}) \\
\texttt{pos} & : & \mathtt{D}((\mathtt{R},\mathtt{R}) \cap \varepsilon, \mathtt{R} \cap \eta, \mathtt{R}) \to ((\mathtt{R},\mathtt{R}) \cap \varepsilon, \mathtt{R} \cap \eta) \cap \mathit{Pos} \\
\texttt{cdn} & : & ((\mathtt{R},\mathtt{R}) \cap \varepsilon, \mathtt{R}) \cap \mathit{Pos} \to (\mathtt{R},\mathtt{R}) \cap \varepsilon \\
\texttt{fst} & : & ((\mathtt{R},\mathtt{R}) \cap \mathit{Coord} \to \mathtt{R}) \cap \\
 & & (\mathit{Cart} \to \mathit{Cx}) \cap (\mathit{Polar} \to \mathit{Radius}) \\
\texttt{snd} & : & ((\mathtt{R},\mathtt{R}) \cap \mathit{Coord} \to \mathtt{R}) \cap \\
 & & (\mathit{Cart} \to \mathit{Cy}) \cap (\mathit{Polar} \to \mathit{Angle}) \\
\texttt{tmp} & : & \mathtt{D}((\mathtt{R},\mathtt{R}), \mathtt{R}, \mathtt{R} \cap \varepsilon) \to \mathtt{R} \cap \varepsilon \\
\texttt{cc2pl} & : & (\mathtt{R},\mathtt{R}) \cap \mathit{Cart} \to (\mathtt{R},\mathtt{R}) \cap \mathit{Polar} \\
\texttt{cl2fh} & : & \mathtt{R} \cap \mathit{Cel} \to \mathtt{R} \cap \mathit{Fh} \\
\} & &
\end{array}
$$

Synthesis via inhabitation

$$
\begin{array}{lcl}
\Gamma = \{ & & \\
\texttt{Tr} & : & () \to \mathtt{D}((\mathtt{R},\mathtt{R}) \cap \mathit{Cart}, \mathtt{R} \cap \mathit{Gpst}, \mathtt{R} \cap \mathit{Cel}) \\
\texttt{pos} & : & \mathtt{D}((\mathtt{R},\mathtt{R}) \cap \varepsilon, \mathtt{R} \cap \eta, \mathtt{R}) \to ((\mathtt{R},\mathtt{R}) \cap \varepsilon, \mathtt{R} \cap \eta) \cap \mathit{Pos} \\
\texttt{cdn} & : & ((\mathtt{R},\mathtt{R}) \cap \varepsilon, \mathtt{R}) \cap \mathit{Pos} \to (\mathtt{R},\mathtt{R}) \cap \varepsilon \\
\texttt{fst} & : & ((\mathtt{R},\mathtt{R}) \cap \mathit{Coord} \to \mathtt{R}) \cap \\
 & & (\mathit{Cart} \to \mathit{Cx}) \cap (\mathit{Polar} \to \mathit{Radius}) \\
\texttt{snd} & : & ((\mathtt{R},\mathtt{R}) \cap \mathit{Coord} \to \mathtt{R}) \cap \\
 & & (\mathit{Cart} \to \mathit{Cy}) \cap (\mathit{Polar} \to \mathit{Angle}) \\
\texttt{tmp} & : & \mathtt{D}((\mathtt{R},\mathtt{R}), \mathtt{R}, \mathtt{R} \cap \varepsilon) \to \mathtt{R} \cap \varepsilon \\
\texttt{cc2pl} & : & (\mathtt{R},\mathtt{R}) \cap \mathit{Cart} \to (\mathtt{R},\mathtt{R}) \cap \mathit{Polar} \\
\texttt{cl2fh} & : & \mathtt{R} \cap \mathit{Cel} \to \mathtt{R} \cap \mathit{Fh} \\
\} & &
\end{array}
$$

$\Gamma \vdash ? : \mathtt{R} \cap \mathit{Fh}$

Synthesis via inhabitation

$$
\begin{array}{lcl}
\Gamma = \{ & & \\
\texttt{Tr} & : & () \to \mathtt{D}((\mathtt{R},\mathtt{R}) \cap \mathit{Cart}, \mathtt{R} \cap \mathit{Gpst}, \mathtt{R} \cap \mathit{Cel}) \\
\texttt{pos} & : & \mathtt{D}((\mathtt{R},\mathtt{R}) \cap \varepsilon, \mathtt{R} \cap \eta, \mathtt{R}) \to ((\mathtt{R},\mathtt{R}) \cap \varepsilon, \mathtt{R} \cap \eta) \cap \mathit{Pos} \\
\texttt{cdn} & : & ((\mathtt{R},\mathtt{R}) \cap \varepsilon, \mathtt{R}) \cap \mathit{Pos} \to (\mathtt{R},\mathtt{R}) \cap \varepsilon \\
\texttt{fst} & : & ((\mathtt{R},\mathtt{R}) \cap \mathit{Coord} \to \mathtt{R}) \cap \\
 & & (\mathit{Cart} \to \mathit{Cx}) \cap (\mathit{Polar} \to \mathit{Radius}) \\
\texttt{snd} & : & ((\mathtt{R},\mathtt{R}) \cap \mathit{Coord} \to \mathtt{R}) \cap \\
 & & (\mathit{Cart} \to \mathit{Cy}) \cap (\mathit{Polar} \to \mathit{Angle}) \\
\texttt{tmp} & : & \mathtt{D}((\mathtt{R},\mathtt{R}), \mathtt{R}, \mathtt{R} \cap \varepsilon) \to \mathtt{R} \cap \varepsilon \\
\texttt{cc2pl} & : & (\mathtt{R},\mathtt{R}) \cap \mathit{Cart} \to (\mathtt{R},\mathtt{R}) \cap \mathit{Polar} \\
\texttt{cl2fh} & : & \mathtt{R} \cap \mathit{Cel} \to \mathtt{R} \cap \mathit{Fh} \\
\} & &
\end{array}
$$

$$
\Gamma \vdash ? : \mathtt{R} \cap \mathit{Fh} \qquad \rightsquigarrow \quad \Gamma \vdash \texttt{cl2fh}\ (\texttt{tmp}\ \texttt{Tr}())) : \mathtt{R} \cap \mathit{Fh}
$$

Synthesis via inhabitation

$$
\begin{array}{lcl}
\Gamma = \{ & & \\
\texttt{Tr} & : & () \to \mathtt{D}((\mathtt{R},\mathtt{R}) \cap \mathit{Cart}, \mathtt{R} \cap \mathit{Gpst}, \mathtt{R} \cap \mathit{Cel}) \\
\texttt{pos} & : & \mathtt{D}((\mathtt{R},\mathtt{R}) \cap \varepsilon, \mathtt{R} \cap \eta, \mathtt{R}) \to ((\mathtt{R},\mathtt{R}) \cap \varepsilon, \mathtt{R} \cap \eta) \cap \mathit{Pos} \\
\texttt{cdn} & : & ((\mathtt{R},\mathtt{R}) \cap \varepsilon, \mathtt{R}) \cap \mathit{Pos} \to (\mathtt{R},\mathtt{R}) \cap \varepsilon \\
\texttt{fst} & : & ((\mathtt{R},\mathtt{R}) \cap \mathit{Coord} \to \mathtt{R}) \cap \\
 & & (\mathit{Cart} \to \mathit{Cx}) \cap (\mathit{Polar} \to \mathit{Radius}) \\
\texttt{snd} & : & ((\mathtt{R},\mathtt{R}) \cap \mathit{Coord} \to \mathtt{R}) \cap \\
 & & (\mathit{Cart} \to \mathit{Cy}) \cap (\mathit{Polar} \to \mathit{Angle}) \\
\texttt{tmp} & : & \mathtt{D}((\mathtt{R},\mathtt{R}), \mathtt{R}, \mathtt{R} \cap \varepsilon) \to \mathtt{R} \cap \varepsilon \\
\texttt{cc2pl} & : & (\mathtt{R},\mathtt{R}) \cap \mathit{Cart} \to (\mathtt{R},\mathtt{R}) \cap \mathit{Polar} \\
\texttt{cl2fh} & : & \mathtt{R} \cap \mathit{Cel} \to \mathtt{R} \cap \mathit{Fh} \\
\} & &
\end{array}
$$

$\Gamma \vdash ? : \mathtt{R} \cap \mathit{Fh} \quad \rightsquigarrow \quad \Gamma \vdash \texttt{cl2fh (tmp Tr())} : \mathtt{R} \cap \mathit{Fh}$

$\Gamma \vdash ? : \mathtt{R} \cap \mathit{Radius}$

Synthesis via inhabitation

$$
\begin{array}{lcl}
\Gamma = \{ & & \\
\texttt{Tr} & : & () \to \mathtt{D}((\mathtt{R},\mathtt{R}) \cap \mathit{Cart}, \mathtt{R} \cap \mathit{Gpst}, \mathtt{R} \cap \mathit{Cel}) \\
\texttt{pos} & : & \mathtt{D}((\mathtt{R},\mathtt{R}) \cap \varepsilon, \mathtt{R} \cap \eta, \mathtt{R}) \to ((\mathtt{R},\mathtt{R}) \cap \varepsilon, \mathtt{R} \cap \eta) \cap \mathit{Pos} \\
\texttt{cdn} & : & ((\mathtt{R},\mathtt{R}) \cap \varepsilon, \mathtt{R}) \cap \mathit{Pos} \to (\mathtt{R},\mathtt{R}) \cap \varepsilon \\
\texttt{fst} & : & ((\mathtt{R},\mathtt{R}) \cap \mathit{Coord} \to \mathtt{R}) \cap \\
 & & (\mathit{Cart} \to \mathit{Cx}) \cap (\mathit{Polar} \to \mathit{Radius}) \\
\texttt{snd} & : & ((\mathtt{R},\mathtt{R}) \cap \mathit{Coord} \to \mathtt{R}) \cap \\
 & & (\mathit{Cart} \to \mathit{Cy}) \cap (\mathit{Polar} \to \mathit{Angle}) \\
\texttt{tmp} & : & \mathtt{D}((\mathtt{R},\mathtt{R}), \mathtt{R}, \mathtt{R} \cap \varepsilon) \to \mathtt{R} \cap \varepsilon \\
\texttt{cc2pl} & : & (\mathtt{R},\mathtt{R}) \cap \mathit{Cart} \to (\mathtt{R},\mathtt{R}) \cap \mathit{Polar} \\
\texttt{cl2fh} & : & \mathtt{R} \cap \mathit{Cel} \to \mathtt{R} \cap \mathit{Fh} \\
\} & &
\end{array}
$$

$$\Gamma \vdash ? : \mathtt{R} \cap \mathit{Fh} \quad \rightsquigarrow \quad \Gamma \vdash \texttt{cl2fh (tmp Tr())} : \mathtt{R} \cap \mathit{Fh}$$

$$\Gamma \vdash ? : \mathtt{R} \cap \mathit{Radius} \quad \rightsquigarrow \quad \Gamma \vdash \texttt{fst(cc2pl(cdn(pos Tr())))} : \mathtt{R} \cap \mathit{Radius}$$

Overview

- Theoretical background
- Composition synthesis
- Examples

Inhabitation in Combinatory Logic

- Usually, in CL, one considers fixed *base* of combinators understood as axiom *schemes*
- Example: **SK**-calculus
  - $\mathbf{K} : \alpha \to \beta \to \alpha$
  - $\mathbf{S} : (\alpha \to (\beta \to \gamma)) \to (\alpha \to \beta) \to (\alpha \to \gamma)$
- Logically, such systems are Hilbert-style proof systems: modus ponens + schematism
- Under Curry-Howard isomorphism, axiom schemes are polymorphic types, and provability is the question of *inhabitation*: $\Gamma \vdash ? : \tau$
- Example: Inhabitation in **SK**-calculus ($\sim$ simple typed λ-calculus) is $\textsc{Pspace}$ complete (Statman 1979).

Synthesis: Relativized Inhabitation

- Since repositories Γ vary and each may change, fixed-base problem is not the right model for synthesis

Synthesis: Relativized Inhabitation

- Since repositories Γ vary and each may change, fixed-base problem is not the right model for synthesis
- **Relativized inhabitation**: the set of combinators is *not fixed* but given as part of input
  - *Given* Γ *and* τ*, does there exist applicative term* e *such that* $\Gamma \vdash e : \tau$*?*

Synthesis: Relativized Inhabitation

- Since repositories Γ vary and each may change, fixed-base problem is not the right model for synthesis
- **Relativized inhabitation**: the set of combinators is *not fixed* but given as part of input
  - *Given* Γ *and* τ*, does there exist applicative term* e *such that* $\Gamma \vdash e : \tau$*?*
- The relativized inhabitation problem for CL is *undecidable*, even in simple types (Linial-Post theorem)

Linial-Post Theorems

- Tarski (Princeton Bicentennial 1946): decision problem (provability) for arbitrary propositional calculi (PPC)?
- L. Linial and E. Post, 1948: there exists an undecidable PPC [*Bulletin of the AMS* (55) 1949]
- Gladstone [1965], Singletary [1967, 1974]: implicational fragment of PPC can represent every r.e. many-one degree
- Later, many theories have been considered in combinatory logic (**B**, **B**′ and friends). Some turned out to be surprisingly complicated (e.g. relevance logic, ticket entailment)

Two-counter Automata

$\mathcal{A} = \langle Q, q_0, q_F, \delta \rangle$

- Control states Q (start state q_0, accepting state q_F)
- Counters $\mathtt{c}_1$ and $\mathtt{c}_2$
- Transition relation δ given by the instructions ($i = 1, 2$)
  - $\mathtt{q} : \mathtt{c}_i := \mathtt{c}_i + 1$; `goto` p
  - $\mathtt{q} : \mathtt{c}_i := \mathtt{c}_i - 1$; `goto` p
  - q : `if` $(\mathtt{c}_i = 0)$ `then goto` p `else goto` r
- Configurations $C = (q, n, m)$, with $q \in Q$ and $n, m \in \mathbb{N}$ values of $\mathtt{c}_1$, $\mathtt{c}_2$

Simple Types are Turing-complete!

For $C = (q, n, m)$, let

$$[C] = (\mathtt{S}(\mathtt{q}) \to \mathtt{C}_1(\mathtt{s}_1^n(0_1)) \to \mathtt{C}_2(\mathtt{s}_2^m(0_2)) \to \mathtt{Acc}$$

Simple Types are Turing-complete!

For $C = (q, n, m)$, let

$$[C] = (\mathtt{S(q)} \to \mathtt{C}_1(\mathtt{s}_1^n(\mathtt{0}_1)) \to \mathtt{C}_2(\mathtt{s}_2^m(\mathtt{0}_2)) \to \mathtt{Acc}$$

$$\begin{array}{lll}
\Gamma_{\mathcal{A}} = \{ & \mathtt{Fin} & : \quad \mathtt{S(q}_F\mathtt{)} \to \mathtt{C}_1(\alpha) \to \mathtt{C}_2(\beta) \to \mathtt{Acc} \\
 & & \\
 & & \bullet\ \mathtt{q} : \mathtt{c}_1 := \mathtt{c}_1 + \mathtt{1};\ \mathtt{goto\ p} \\
 & \mathtt{Add}_1[\mathtt{q},\mathtt{p}] & : \quad (\mathtt{S(p)} \to \mathtt{C}_1(\mathtt{s}_1(\alpha)) \to \mathtt{C}_2(\beta) \to \mathtt{Acc}) \to \\
 & & \quad\ (\mathtt{S(q)} \to \mathtt{C}_1(\alpha) \to \mathtt{C}_2(\beta) \to \mathtt{Acc}) \\
 & & \\
 & & \bullet\ \mathtt{q} : \mathtt{c}_1 := \mathtt{c}_1 - \mathtt{1};\ \mathtt{goto\ p} \\
 & \mathtt{Sub}_1[\mathtt{q},\mathtt{p}] & : \quad (\mathtt{S(p)} \to \mathtt{C}_1(\alpha) \to \mathtt{C}_2(\beta) \to \mathtt{Acc}) \to \\
 & & \quad\ (\mathtt{S(q)} \to \mathtt{C}_1(\mathtt{s}_1(\alpha)) \to \mathtt{C}_2(\beta) \to \mathtt{Acc}) \\
 & & \\
 & & \bullet\ \mathtt{q} : \mathtt{if}\ (\mathtt{c}_1 = \mathtt{0})\ \mathtt{then\ goto\ p\ else\ goto\ r} \\
 & \mathtt{Tst}_1^{\mathtt{Z}}[\mathtt{q},\mathtt{p}] & : \quad (\mathtt{S(p)} \to \mathtt{C}_1(\mathtt{0}_1) \to \mathtt{C}_2(\beta) \to \mathtt{Acc}) \to \\
 & & \quad\ (\mathtt{S(q)} \to \mathtt{C}_1(\mathtt{0}_1) \to \mathtt{C}_2(\beta) \to \mathtt{Acc}) \\
 & & \\
 & \mathtt{Tst}_1^{\mathtt{NZ}}[\mathtt{q},\mathtt{r}] & : \quad (\mathtt{S(r)} \to \mathtt{C}_1(\mathtt{s}_1(\alpha)) \to \mathtt{C}_2(\beta) \to \mathtt{Acc}) \to \\
 & & \quad\ (\mathtt{S(q)} \to \mathtt{C}_1(\mathtt{s}_1(\alpha)) \to \mathtt{C}_2(\beta) \to \mathtt{Acc}) \\
\}
\end{array}$$

Simple Types are Turing-complete!

For $C = (q, n, m)$, let

$$[C] = (\mathtt{S(q)} \to \mathtt{C}_1(\mathtt{s}_1^n(\mathtt{0}_1)) \to \mathtt{C}_2(\mathtt{s}_2^m(\mathtt{0}_2)) \to \mathtt{Acc}$$

$$\begin{array}{lll}
\Gamma_{\mathcal{A}} = \{ & \mathtt{Fin} & : \quad \mathtt{S(q}_F\mathtt{)} \to \mathtt{C}_1(\alpha) \to \mathtt{C}_2(\beta) \to \mathtt{Acc} \\
 & & \\
 & & \bullet\ \mathtt{q} : \mathtt{c}_1 := \mathtt{c}_1 + \mathtt{1};\ \mathtt{goto\ p} \\
 & \mathtt{Add}_1[\mathtt{q},\mathtt{p}] & : \quad (\mathtt{S(p)} \to \mathtt{C}_1(\mathtt{s}_1(\alpha)) \to \mathtt{C}_2(\beta) \to \mathtt{Acc}) \to \\
 & & \quad\ (\mathtt{S(q)} \to \mathtt{C}_1(\alpha) \to \mathtt{C}_2(\beta) \to \mathtt{Acc}) \\
 & & \\
 & & \bullet\ \mathtt{q} : \mathtt{c}_1 := \mathtt{c}_1 - \mathtt{1};\ \mathtt{goto\ p} \\
 & \mathtt{Sub}_1[\mathtt{q},\mathtt{p}] & : \quad (\mathtt{S(p)} \to \mathtt{C}_1(\alpha) \to \mathtt{C}_2(\beta) \to \mathtt{Acc}) \to \\
 & & \quad\ (\mathtt{S(q)} \to \mathtt{C}_1(\mathtt{s}_1(\alpha)) \to \mathtt{C}_2(\beta) \to \mathtt{Acc}) \\
 & & \\
 & & \bullet\ \mathtt{q} : \mathtt{if}\ (\mathtt{c}_1 = \mathtt{0})\ \mathtt{then\ goto\ p\ else\ goto\ r} \\
 & \mathtt{Tst}_1^{\mathtt{Z}}[\mathtt{q},\mathtt{p}] & : \quad (\mathtt{S(p)} \to \mathtt{C}_1(\mathtt{0}_1) \to \mathtt{C}_2(\beta) \to \mathtt{Acc}) \to \\
 & & \quad\ (\mathtt{S(q)} \to \mathtt{C}_1(\mathtt{0}_1) \to \mathtt{C}_2(\beta) \to \mathtt{Acc}) \\
 & & \\
 & \mathtt{Tst}_1^{\mathtt{NZ}}[\mathtt{q},\mathtt{r}] & : \quad (\mathtt{S(r)} \to \mathtt{C}_1(\mathtt{s}_1(\alpha)) \to \mathtt{C}_2(\beta) \to \mathtt{Acc}) \to \\
 & & \quad\ (\mathtt{S(q)} \to \mathtt{C}_1(\mathtt{s}_1(\alpha)) \to \mathtt{C}_2(\beta) \to \mathtt{Acc}) \\
\}
\end{array}$$

Theorem $\mathcal{A}$ *accepts from* $C \Leftrightarrow \exists e.\ \Gamma_{\mathcal{A}} \vdash e : [C]$

Inhabitation as Logic Programming

- The input theory Γ is the program and search for inhabitants is its execution [4]

[4] Broadly related: [Miller,Nadathur,Pfenning,Scedrov: *Uniform proofs as a foundation for logic programming*, Ann. Pure and Appl. Logic 1991]

Inhabitation as Logic Programming

- The input theory Γ is the program and search for inhabitants is its execution [4]
  - *the input repository* Γ *is a logic program (or "interface program") at the level of types*

[4] Broadly related: [Miller,Nadathur,Pfenning,Scedrov: *Uniform proofs as a foundation for logic programming*, Ann. Pure and Appl. Logic 1991]

Inhabitation as Logic Programming

- The input theory Γ is the program and search for inhabitants is its execution [4]
  - *the input repository* Γ *is a logic program (or "interface program") at the level of types*
  - *the interface types of each component combinator in* Γ *are rules in this program*

[4] Broadly related: [Miller,Nadathur,Pfenning,Scedrov: *Uniform proofs as a foundation for logic programming*, Ann. Pure and Appl. Logic 1991]

Inhabitation as Logic Programming

- The input theory Γ is the program and search for inhabitants is its execution [4]
  - *the input repository* Γ *is a logic program (or "interface program") at the level of types*
  - *the interface types of each component combinator in* Γ *are rules in this program*
  - *the inhabitation goal* τ *is the input goal to the program*

[4] Broadly related: [Miller,Nadathur,Pfenning,Scedrov: *Uniform proofs as a foundation for logic programming*, Ann. Pure and Appl. Logic 1991]

Inhabitation as Logic Programming

- The input theory Γ is the program and search for inhabitants is its execution [4]
  - *the input repository* Γ *is a logic program (or "interface program") at the level of types*
  - *the interface types of each component combinator in* Γ *are rules in this program*
  - *the inhabitation goal* τ *is the input goal to the program*
  - *the search for inhabitants is the execution of the program*

[4] Broadly related: [Miller,Nadathur,Pfenning,Scedrov: *Uniform proofs as a foundation for logic programming*, Ann. Pure and Appl. Logic 1991]

Inhabitation as Logic Programming

- The input theory Γ is the program and search for inhabitants is its execution [4]
  - *the input repository* Γ *is a logic program (or "interface program") at the level of types*
  - *the interface types of each component combinator in* Γ *are rules in this program*
  - *the inhabitation goal* τ *is the input goal to the program*
  - *the search for inhabitants is the execution of the program*
  - *the inhabitants are programs generated by synthesis from the interface program as its solution space*

[4] Broadly related: [Miller,Nadathur,Pfenning,Scedrov: *Uniform proofs as a foundation for logic programming*, Ann. Pure and Appl. Logic 1991]

Finite and bounded CL

- Algorithms, complexity, expressive power of relativized inhabitation with and w/o intersection types
- Connection to computational models: tree automata and alternating Turing machines
  - In distinction to standard CL, we consider the *relativized* inhabitation question, but we bound it
  - In *finite* CL (FCL) we eliminate schematism (monomorphic restriction)
  - In *k-bounded* CL (BCL_k) we bound the depth of types in scheme instantiation (depth k-bounded polymorphism)

Finite and bounded CL

Theorem[Rehof,Urzyczyn TLCA 2011] *For finite combinatory logic we have:*

1. *Relativized inhabitation in* $\textsc{fcl}(\to)$ *is in* $\textsc{Ptime}$
2. *Relativized inhabitation in* $\textsc{fcl}(\to,\cap)$ *is* $\textsc{Exptime}$*-complete*

Theorem[Düdder,Martens,Rehof,Urzyczyn CSL 2012] *For bounded combinatory logic we have:*

1. *Relativized inhabitation in* $\textsc{bcl}_k(\to)$ *is* $\textsc{Exptime}$*-complete for all* k
2. *Relativized inhabitation in* $\textsc{bcl}_k(\to,\cap)$ *is* $(k+2)$-$\textsc{Exptime}$*-complete*

Bounded combinatory logic

- Levels

$$\begin{array}{lcl} \ell(a) & = & 0, \text{ for } a \in \mathbb{A} \cup \mathbb{V}; \\ \ell(\tau \to \sigma) & = & 1 + \max\{\ell(\tau), \ell(\sigma)\}; \\ \ell(\bigcap_{i=1}^{n} \tau_i) & = & \max\{\ell(\tau_i) \mid i = 1, \ldots, n\}. \end{array}$$

Bounded combinatory logic

- Levels

$$\begin{array}{lcl} \ell(a) & = & 0, \text{ for } a \in \mathbb{A} \cup \mathbb{V}; \\ \ell(\tau \to \sigma) & = & 1 + \max\{\ell(\tau), \ell(\sigma)\}; \\ \ell(\bigcap_{i=1}^{n} \tau_i) & = & \max\{\ell(\tau_i) \mid i = 1, \ldots, n\}. \end{array}$$

- $\mathrm{BCL}_k(\to, \cap)$

$$\frac{[\ell(S) \leq k]}{\Gamma, x : \tau \vdash_k x : S(\tau)}(\mathsf{var}) \qquad \frac{\Gamma \vdash_k e : \tau \to \tau' \quad \Gamma \vdash_k e' : \tau}{\Gamma \vdash_k (e\ e') : \tau'}(\to\mathsf{E})$$

$$\frac{\Gamma \vdash_k e : \tau_1 \quad \Gamma \vdash_k e : \tau_2}{\Gamma \vdash_k e : \tau_1 \cap \tau_2}(\cap\mathsf{I}) \qquad \frac{\Gamma \vdash_k e : \tau \quad \tau \leq \tau'}{\Gamma \vdash_k e : \tau'}(\leq)$$

Subtyping

$$\sigma \leq \boldsymbol{\omega}, \quad \boldsymbol{\omega} \leq \boldsymbol{\omega} \to \boldsymbol{\omega}, \quad \sigma \cap \tau \leq \sigma, \quad \sigma \cap \tau \leq \tau, \quad \sigma \leq \sigma \cap \sigma;$$
$$(\sigma \to \tau) \cap (\sigma \to \rho) \leq \sigma \to \tau \cap \rho;$$

If $\sigma \leq \sigma'$ and $\tau \leq \tau'$ then $\sigma \cap \tau \leq \sigma' \cap \tau'$ and $\sigma' \to \tau \leq \sigma \to \tau'$

Derived distributivity properties:

$$(\sigma \to \tau) \cap (\sigma \to \rho) = \sigma \to (\tau \cap \rho)$$
$$(\sigma \to \tau) \cap (\sigma' \to \tau') \leq (\sigma \cap \sigma') \to (\tau \cap \tau')$$

Upper bound - ATM algorithm

Input : Γ, τ, k

$$\Gamma = \{f : (0 \to 1) \cap (1 \to 0),\\ x : (\alpha \to \beta) \to (\beta \to \gamma) \to (\alpha \to \gamma)\}$$
$$\tau = (0 \to 0) \cap (1 \to 1)$$

Upper bound - ATM algorithm

Input : Γ, τ, k

loop :

1 CHOOSE $(x : \sigma) \in \Gamma$;

2 $\sigma' := \bigcap\{S(\sigma) \mid S \in \mathcal{S}_x^{(\Gamma,\tau,k)}\}$;

$\Gamma = \{f : (0 \to 1) \cap (1 \to 0),$
$\quad x : (\alpha \to \beta) \to (\beta \to \gamma) \to (\alpha \to \gamma)\}$
$\tau = (0 \to 0) \cap (1 \to 1)$

$\sigma' = (0 \to 0) \to (0 \to 0) \to (0 \to 0) \cap \cdots \cap$
$\quad (1 \to 1) \to (1 \to 1) \to (1 \to 1)$

Upper bound - ATM algorithm

Input : Γ, τ, k

$\Gamma = \{f : (0 \to 1) \cap (1 \to 0),$
$\quad x : (\alpha \to \beta) \to (\beta \to \gamma) \to (\alpha \to \gamma)\}$
$\tau = (0 \to 0) \cap (1 \to 1)$

loop :

1 CHOOSE $(x : \sigma) \in \Gamma$;

2 $\sigma' := \bigcap\{S(\sigma) \mid S \in \mathcal{S}_x^{(\Gamma,\tau,k)}\}$;

$\sigma' = (0 \to 0) \to (0 \to 0) \to (0 \to 0) \cap \cdots \cap$
$\quad (1 \to 1) \to (1 \to 1) \to (1 \to 1)$

3 CHOOSE $m \in \{0, \ldots, \|\sigma'\|\}$;

4 CHOOSE $P \subseteq \mathbb{P}_m(\sigma')$;

$(0 \to 1) \to (1 \to 0) \to (0 \to 0) \cap$
$(1 \to 0) \to (0 \to 1) \to (1 \to 1)$

Upper bound - ATM algorithm

Input : Γ, τ, k

$\Gamma = \{f : (0 \to 1) \cap (1 \to 0),$
$x : (\alpha \to \beta) \to (\beta \to \gamma) \to (\alpha \to \gamma)\}$
$\tau = (0 \to 0) \cap (1 \to 1)$

loop :

1 CHOOSE $(x : \sigma) \in \Gamma$;

2 $\sigma' := \bigcap\{S(\sigma) \mid S \in \mathcal{S}_x^{(\Gamma,\tau,k)}\}$;

$\sigma' = (0 \to 0) \to (0 \to 0) \to (0 \to 0) \cap \cdots \cap$
$(1 \to 1) \to (1 \to 1) \to (1 \to 1)$

3 CHOOSE $m \in \{0, \ldots, \|\sigma'\|\}$;

4 CHOOSE $P \subseteq \mathbb{P}_m(\sigma')$;

$(0 \to 1) \to (1 \to 0) \to (0 \to 0) \cap$
$(1 \to 0) \to (0 \to 1) \to (1 \to 1)$

5 IF $(\bigcap_{\pi \in P} tgt_m(\pi) \leq \tau)$ THEN

6 IF $(m = 0)$ THEN ACCEPT;

$(0 \to 0) \cap (1 \to 1) \leq \tau$

Upper bound - ATM algorithm

Input : Γ, τ, k

$\Gamma = \{f : (0 \to 1) \cap (1 \to 0),$
$x : (\alpha \to \beta) \to (\beta \to \gamma) \to (\alpha \to \gamma)\}$
$\tau = (0 \to 0) \cap (1 \to 1)$

loop :

1 CHOOSE $(x : \sigma) \in \Gamma$;

2 $\sigma' := \bigcap\{S(\sigma) \mid S \in \mathcal{S}_x^{(\Gamma,\tau,k)}\}$;

$\sigma' = (0 \to 0) \to (0 \to 0) \to (0 \to 0) \cap \cdots \cap$
$(1 \to 1) \to (1 \to 1) \to (1 \to 1)$

3 CHOOSE $m \in \{0, \ldots, \|\sigma'\|\}$;

4 CHOOSE $P \subseteq \mathbb{P}_m(\sigma')$;

$(0 \to 1) \to (1 \to 0) \to (0 \to 0) \cap$
$(1 \to 0) \to (0 \to 1) \to (1 \to 1)$

5 IF $(\bigcap_{\pi \in P} tgt_m(\pi) \leq \tau)$ THEN

$(0 \to 0) \cap (1 \to 1) \leq \tau$

6 IF $(m = 0)$ THEN ACCEPT;

7 ELSE

8 FORALL$(i = 1 \ldots m)$

9 $\tau := \bigcap_{\pi \in P} arg_i(\pi)$;

$\tau := (0 \to 1) \cap (1 \to 0)$
$\tau := (1 \to 0) \cap (0 \to 1)$

10 GOTO *loop*;

11 ELSE REJECT;

Upper bound - ATM algorithm

Input : Γ, τ, k

$\Gamma = \{f : (0 \to 1) \cap (1 \to 0),$
$x : (\alpha \to \beta) \to (\beta \to \gamma) \to (\alpha \to \gamma)\}$
$\tau = (0 \to 0) \cap (1 \to 1)$

loop :

1 CHOOSE $(x : \sigma) \in \Gamma$;

2 $\sigma' := \bigcap\{S(\sigma) \mid S \in \mathcal{S}_x^{(\Gamma,\tau,k)}\}$;

$\sigma' = (0 \to 0) \to (0 \to 0) \to (0 \to 0) \cap \cdots \cap$
$(1 \to 1) \to (1 \to 1) \to (1 \to 1)$

3 CHOOSE $m \in \{0, \ldots, \|\sigma'\|\}$;

4 CHOOSE $P \subseteq \mathbb{P}_m(\sigma')$;

$(0 \to 1) \to (1 \to 0) \to (0 \to 0) \cap$
$(1 \to 0) \to (0 \to 1) \to (1 \to 1)$

5 IF $(\bigcap_{\pi \in P} tgt_m(\pi) \leq \tau)$ THEN

$(0 \to 0) \cap (1 \to 1) \leq \tau$

6 IF $(m = 0)$ THEN ACCEPT;

7 ELSE

8 FORALL$(i = 1 \ldots m)$

9 $\tau := \bigcap_{\pi \in P} arg_i(\pi)$;

$\tau := (0 \to 1) \cap (1 \to 0)$
$\tau := (1 \to 0) \cap (0 \to 1)$

10 GOTO *loop*;

11 ELSE REJECT;

$(x\ f)\ f : (0 \to 0) \cap (1 \to 1)$

Overview

- Theoretical background
- Composition synthesis
- Examples

Situation

- Implemented components in *native language*

```
F ≜
letrec Q : [int] → [int] =
  λx : [int]. if (x = nil) then nil
  else
       (Q (Filter (tail x) (λy: int.(head x) ≤ y))) ::
       (head x) ::
       (Q (Filter (tail x) (λy: int.(head x) > y)))
```

  with *native type*, e.g., $[\texttt{int}] \rightarrow [\texttt{int}]$

- Exposed in semantic repository as combinator $(\texttt{F} : \tau)$ with *semantic type*, e.g.,

$$\tau \quad = \quad ([\texttt{int}] \rightarrow ([\texttt{int}] \cap \mathit{Sorted})) \cap \mathit{SortingFunction.Quicksort}$$

Native vs. Semantic Correctness

- *Native* correctness: for each combinatory interface $(\mathtt{F} : \tau)$ a unique underlying native type $A = \tau^\circ$ and $\Gamma \Vdash e : \tau$, where

$$\Gamma \Vdash e : \tau \quad \textit{if and only if} \quad \Gamma \vdash e : \tau \quad \textit{and} \quad \Gamma^\circ \vdash_s e : \tau^\circ$$

- *Semantic* correctness: *assume* (but do not prove) that semantic types correctly describe semantics of component implementation [5]

[5] [Haack,Howard,Stoughton,Wells: *Fully automatic adaptation of software components based on semantic specifications*, AMAST 2002]

Stratified System

Types

$$\begin{array}{lcll} \mathbb{T}_N & \ni & A, B & ::= \quad \alpha \mid c \mid A \to B \\ & & & \qquad\qquad \alpha, \beta \in \mathbb{V}_N, c \in \mathbb{C}_N \end{array}$$

$$\begin{array}{lcll} \mathbb{T}_S & \ni & \varphi, \psi & ::= \quad \varepsilon \mid d \mid \omega \mid \varphi \to \psi \mid \varphi \cap \psi \\ & & & \qquad\qquad \varepsilon, \eta \in \mathbb{V}_S, d \in \mathbb{C}_S \end{array}$$

$$\begin{array}{lcll} \mathbb{T}_U & \ni & \tau, \sigma & ::= \quad A \mid \tau \to \sigma \mid \tau \cap \varphi \end{array}$$

Underlying native types

$$\begin{array}{lcl} A^\circ & = & A \\ (\tau \to \sigma)^\circ & = & \tau^\circ \to \sigma^\circ \\ (\tau \cap \varphi)^\circ & = & \tau^\circ \end{array}$$

(CL)S

Implementation (B. Düdder & M. Martens, Dortmund):
Combinatory **L**ogic **S**ynthesizer

- Inhabitation algorithms for $\vdash_k$ and $\Vdash_k$
- Integrated into larger framework for specification of semantic repositories and synthesis
- Many optimizations done and more underway
- Experiments
- Deployment on cloud platform (Dortmund) with 2000 cores

Overview

- Theoretical background
- Composition synthesis
- Examples

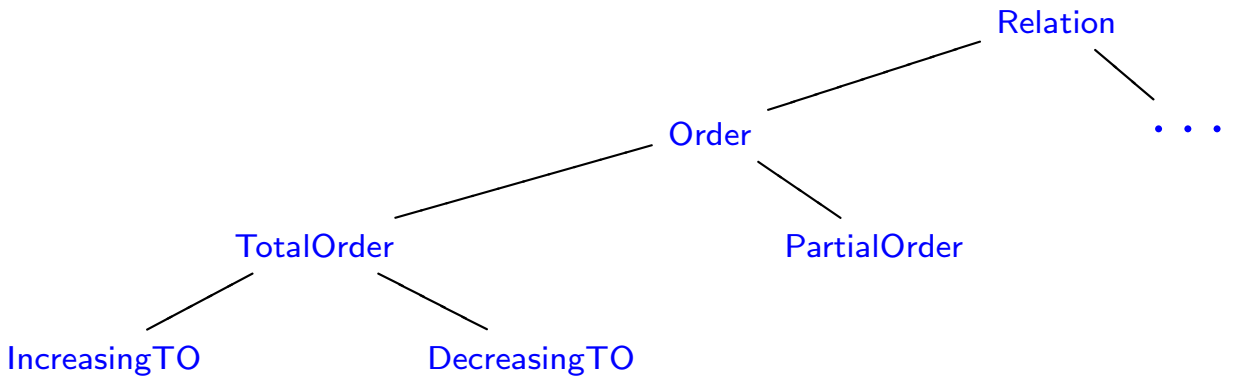
Parameterization

$$
\begin{array}{ll}
\Gamma = \{ & \\
\quad \texttt{F} & : \quad ([\alpha] \to (\alpha \to \texttt{bool}) \to [\alpha]) \cap \mathit{Filter} \\
\quad \texttt{S} & : \quad (([\alpha] \to (\alpha \to \texttt{bool}) \to [\alpha]) \cap \mathit{Filter}) \to \\
 & \qquad ((\alpha \to \alpha \to \texttt{bool}) \to [\alpha] \to [\alpha]) \cap \\
 & \qquad (\mathit{TotalOrder} \to \boldsymbol{\omega} \to \mathit{Sorted}) \cap \\
 & \qquad (\mathit{PartialOrder} \to \boldsymbol{\omega} \to \mathit{TopSorted})) \\
\quad \texttt{G} & : \quad \texttt{Graph}(\alpha) \to ((\alpha \to \alpha \to \texttt{bool}) \cap \mathit{PartialOrder}) \\
\quad \texttt{N} & : \quad \texttt{Graph}(\alpha) \to [\alpha] \\
\quad \circ & : \quad (\alpha \to \beta) \to (\beta \to \gamma) \to (\alpha \to \gamma) \\
\quad \diamond & : \quad (\alpha \to \beta \to \gamma) \to (\alpha \to \beta) \to (\alpha \to \gamma) \\
\} &
\end{array}
$$

$\diamond = \mathbf{S} = \lambda x.\lambda y.\lambda z.xz(yz)$

$\circ = \mathbf{B}' = \lambda x.\lambda y.\lambda z.y(xz)$

Semantic Structures



Parameterization

$$
\begin{array}{ll}
\Gamma = \{ & \\
\quad \texttt{F} & : \quad ([\alpha] \to (\alpha \to \texttt{bool}) \to [\alpha]) \cap \mathit{Filter} \\
\quad \texttt{S} & : \quad (([\alpha] \to (\alpha \to \texttt{bool}) \to [\alpha]) \cap \mathit{Filter}) \to \\
 & \qquad ((\alpha \to \alpha \to \texttt{bool}) \to [\alpha] \to [\alpha]) \cap \\
 & \qquad (\mathit{TotalOrder} \to \boldsymbol{\omega} \to \mathit{Sorted}) \cap \\
 & \qquad (\mathit{PartialOrder} \to \boldsymbol{\omega} \to \mathit{TopSorted})) \\
\quad \texttt{G} & : \quad \texttt{Graph}(\alpha) \to ((\alpha \to \alpha \to \texttt{bool}) \cap \mathit{PartialOrder}) \\
\quad \texttt{N} & : \quad \texttt{Graph}(\alpha) \to [\alpha] \\
\quad \circ & : \quad (\alpha \to \beta) \to (\beta \to \gamma) \to (\alpha \to \gamma) \\
\quad \diamond & : \quad (\alpha \to \beta \to \gamma) \to (\alpha \to \beta) \to (\alpha \to \gamma) \\
\} &
\end{array}
$$

Parameterization

$$
\begin{array}{l}
\Gamma = \{ \\
\quad \texttt{F} \quad : \quad ([\alpha] \to (\alpha \to \texttt{bool}) \to [\alpha]) \cap \mathit{Filter} \\
\quad \texttt{S} \quad : \quad (([\alpha] \to (\alpha \to \texttt{bool}) \to [\alpha]) \cap \mathit{Filter}) \to \\
\qquad\qquad ((\alpha \to \alpha \to \texttt{bool}) \to [\alpha] \to [\alpha]) \cap \\
\qquad\qquad (\mathit{TotalOrder} \to \boldsymbol{\omega} \to \mathit{Sorted}) \cap \\
\qquad\qquad (\mathit{PartialOrder} \to \boldsymbol{\omega} \to \mathit{TopSorted})) \\
\quad \texttt{G} \quad : \quad \texttt{Graph}(\alpha) \to ((\alpha \to \alpha \to \texttt{bool}) \cap \mathit{PartialOrder}) \\
\quad \texttt{N} \quad : \quad \texttt{Graph}(\alpha) \to [\alpha] \\
\quad \circ \quad : \quad (\alpha \to \beta) \to (\beta \to \gamma) \to (\alpha \to \gamma) \\
\quad \diamond \quad : \quad (\alpha \to \beta \to \gamma) \to (\alpha \to \beta) \to (\alpha \to \gamma) \\
\}
\end{array}
$$

Inhabitation goal

$$\Gamma \Vdash ? : \texttt{Graph}(\alpha) \to ([\alpha] \cap \mathit{TopSorted})$$

Parameterization

$$
\begin{array}{l}
\Gamma = \{ \\
\quad \texttt{F} \quad : \quad ([\alpha] \to (\alpha \to \texttt{bool}) \to [\alpha]) \cap \mathit{Filter} \\
\quad \texttt{S} \quad : \quad (([\alpha] \to (\alpha \to \texttt{bool}) \to [\alpha]) \cap \mathit{Filter}) \to \\
\qquad\qquad ((\alpha \to \alpha \to \texttt{bool}) \to [\alpha] \to [\alpha]) \cap \\
\qquad\qquad (\mathit{TotalOrder} \to \boldsymbol{\omega} \to \mathit{Sorted}) \cap \\
\qquad\qquad (\mathit{PartialOrder} \to \boldsymbol{\omega} \to \mathit{TopSorted})) \\
\quad \texttt{G} \quad : \quad \texttt{Graph}(\alpha) \to ((\alpha \to \alpha \to \texttt{bool}) \cap \mathit{PartialOrder}) \\
\quad \texttt{N} \quad : \quad \texttt{Graph}(\alpha) \to [\alpha] \\
\quad \circ \quad : \quad (\alpha \to \beta) \to (\beta \to \gamma) \to (\alpha \to \gamma) \\
\quad \diamond \quad : \quad (\alpha \to \beta \to \gamma) \to (\alpha \to \beta) \to (\alpha \to \gamma) \\
\}
\end{array}
$$

Inhabitation goal

$$\Gamma \Vdash ? : \texttt{Graph}(\alpha) \to ([\alpha] \cap \mathit{TopSorted})$$

with solution

$$\Gamma \Vdash (\texttt{G} \circ (\texttt{S}\ \texttt{F})) \diamond \texttt{N} : \texttt{Graph}(\alpha) \to ([\alpha] \cap \mathit{TopSorted})$$

Service Composition

$$
\begin{array}{ll}
\Gamma = \{ & \\
\quad \texttt{F} & : \quad ([\alpha] \to (\alpha \to \texttt{bool}) \to [\alpha]) \cap \mathit{Filter} \\
\quad \texttt{S} & : \quad (([\alpha] \to (\alpha \to \texttt{bool}) \to [\alpha]) \cap \mathit{Filter}) \to \\
 & \qquad ((\alpha \to \alpha \to \texttt{bool}) \to [\alpha] \to [\alpha]) \cap \\
 & \qquad (\mathit{TotalOrder} \to \boldsymbol{\omega} \to \mathit{Sorted}) \cap \\
 & \qquad (\mathit{PartialOrder} \to \boldsymbol{\omega} \to \mathit{TopSorted})) \\
\quad \texttt{G} & : \quad \texttt{Graph}(\alpha) \to ((\alpha \to \alpha \to \texttt{bool}) \cap \mathit{PartialOrder}) \\
\quad \texttt{N} & : \quad \texttt{Graph}(\alpha) \to [\alpha] \\
\quad \circ & : \quad (\alpha \to \beta) \to (\beta \to \gamma) \to (\alpha \to \gamma) \\
\quad \diamond & : \quad (\alpha \to \beta \to \gamma) \to (\alpha \to \beta) \to (\alpha \to \gamma) \\
\quad \texttt{MyID} & : \quad () \to (\texttt{int} \cap \mathit{UserID}) \\
\quad \texttt{GetTasks} & : \quad () \to \texttt{Graph}(\texttt{Task}) \\
\} &
\end{array}
$$

$$
\begin{array}{ll}
\Delta = \{ & \\
\quad \texttt{S} :: \texttt{Connect} & : \quad () \to (\texttt{int} \cap \mathit{SessionID}) \\
\quad \texttt{S} :: \texttt{ReqTransaction} & : \quad (\texttt{int} \cap \mathit{UserID}) \to (\texttt{int} \cap \mathit{SessionID}) \to \\
 & \qquad ([\texttt{Task}] \cap \mathit{TopSorted}) \to (\texttt{int} \cap \mathit{TID}) \\
\quad \texttt{S} :: \texttt{EndTransaction} & : \quad (\texttt{int} \cap \mathit{TID}) \to [\texttt{int} \cap \mathit{Result}] \\
\} &
\end{array}
$$

Service Composition

We can now ask for the computation of a result list by the inhabitation goal

$$\Gamma \cup \Delta \vdash ? : [\texttt{int} \cap \mathit{Result}]$$

Service Composition

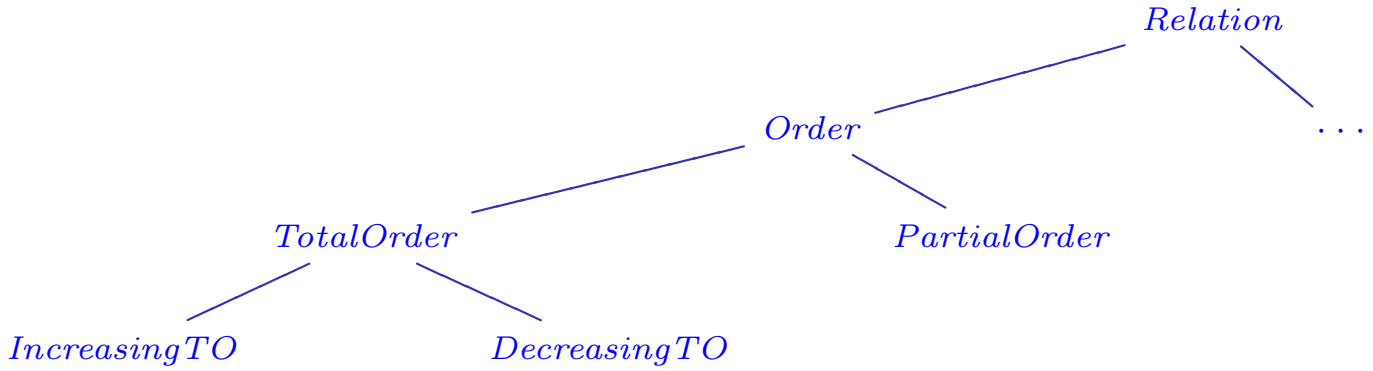
We can now ask for the computation of a result list by the inhabitation goal

$$\Gamma \cup \Delta \Vdash ? : [\texttt{int} \cap \mathit{Result}]$$

which generates the solution

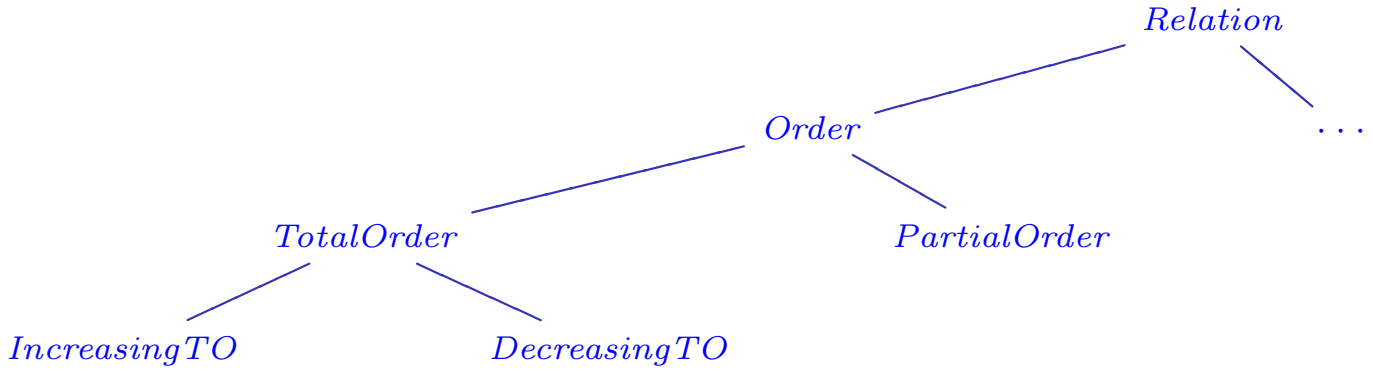
$$\begin{array}{l}\Gamma \cup \Delta \Vdash \texttt{let X} = \texttt{S} :: \texttt{ReqTransaction} \\ \qquad\qquad\qquad\qquad\qquad\qquad (\texttt{MyID}()) \\ \qquad\qquad\qquad\qquad\qquad\qquad\quad (\texttt{S} :: \texttt{Connect}()) \\ \qquad\qquad \texttt{let Y} = (\texttt{G} \circ (\texttt{S F})) \diamond \texttt{N} \\ \qquad\qquad\quad \texttt{in} \quad (\texttt{X} \circ \texttt{Y} \circ \texttt{EndTransaction})\ (\texttt{GetTasks}()) \\ \qquad\qquad : [\texttt{int} \cap \mathit{Result}]\end{array}$$

Logic



- $Rev(R) = \{r^{-1} \mid r \in R\}, r^{-1} = \{(y, x) \mid (x, y) \in r\}$

Logic



- $Rev(R) = \{r^{-1} \mid r \in R\}, r^{-1} = \{(y, x) \mid (x, y) \in r\}$
  - $Rev(Relation.DecreasingTO) = Relation.IncreasingTO$
  - $Rev(Relation.IncreasingTO) = Relation.DecreasingTO$

Logic

$$
\begin{array}{lcl}
\Gamma = \{ & & \\
\quad \texttt{F} & : & ([\alpha] \to (\alpha \to \texttt{bool}) \to [\alpha]) \cap \mathit{Filter} \\
\quad \texttt{S} & : & (([\alpha] \to (\alpha \to \texttt{bool}) \to [\alpha]) \cap \mathit{Filter}) \to \\
 & & \quad ((\alpha \to \alpha \to \texttt{bool}) \to [\alpha] \to [\alpha]) \cap \\
 & & \quad (\mathit{TotalOrder}.\varepsilon \to \boldsymbol{\omega} \to \mathit{Sorted}(\varepsilon)) \cap \\
 & & \quad (\mathit{PartialOrder}.\eta \to \boldsymbol{\omega} \to \mathit{TopSorted}(\eta))) \\
\quad \ldots & & \\
\quad \texttt{swap} & : & ((\alpha \to \beta \to \gamma) \to (\beta \to \alpha \to \gamma)) \cap \\
 & & (\mathit{Relation}.\varepsilon \to \mathit{Rev}(\mathit{Relation}.\varepsilon)) \\
\quad \leq_{\mathbb{N}} & : & (\texttt{int} \to \texttt{int} \to \texttt{bool}) \cap \mathit{Relation.IncreasingTO} \\
\quad \Phi_{\texttt{Rev}} & : & (\alpha \to \alpha) \cap \\
 & & (\mathit{Rev}(\mathit{Relation.IncreasingTO}) \to \mathit{Relation.DecreasingTO}) \cap \\
 & & (\mathit{Rev}(\mathit{Relation.DecreasingTO}) \to \mathit{Relation.IncreasingTO}) \\
\} & &
\end{array}
$$

Logic

$$
\begin{array}{lll}
\Gamma = \{ & & \\
\quad \texttt{F} & : & ([\alpha] \to (\alpha \to \texttt{bool}) \to [\alpha]) \cap \mathit{Filter} \\
\quad \texttt{S} & : & (([\alpha] \to (\alpha \to \texttt{bool}) \to [\alpha]) \cap \mathit{Filter}) \to \\
 & & \quad ((\alpha \to \alpha \to \texttt{bool}) \to [\alpha] \to [\alpha]) \cap \\
 & & \quad (\mathit{TotalOrder}.\varepsilon \to \boldsymbol{\omega} \to \mathit{Sorted}(\varepsilon)) \cap \\
 & & \quad (\mathit{PartialOrder}.\eta \to \boldsymbol{\omega} \to \mathit{TopSorted}(\eta))) \\
\quad \ldots & & \\
\quad \texttt{swap} & : & ((\alpha \to \beta \to \gamma) \to (\beta \to \alpha \to \gamma)) \cap \\
 & & (\mathit{Relation}.\varepsilon \to \mathit{Rev}(\mathit{Relation}.\varepsilon)) \\
\quad \leq_{\mathbb{N}} & : & (\texttt{int} \to \texttt{int} \to \texttt{bool}) \cap \mathit{Relation.IncreasingTO} \\
\quad \Phi_{\texttt{Rev}} & : & (\alpha \to \alpha) \cap \\
 & & (\mathit{Rev}(\mathit{Relation.IncreasingTO}) \to \mathit{Relation.DecreasingTO}) \cap \\
 & & (\mathit{Rev}(\mathit{Relation.DecreasingTO}) \to \mathit{Relation.IncreasingTO}) \\
\} & &
\end{array}
$$

Inhabitation goal

$$\Gamma \Vdash ? : [\texttt{int}] \to ([\texttt{int}] \cap \mathit{Sorted}(\mathit{DecreasingTO}))$$

Logic

$$
\begin{array}{lll}
\Gamma = \{ & & \\
\quad \texttt{F} & : & ([\alpha] \to (\alpha \to \texttt{bool}) \to [\alpha]) \cap \mathit{Filter} \\
\quad \texttt{S} & : & (([\alpha] \to (\alpha \to \texttt{bool}) \to [\alpha]) \cap \mathit{Filter}) \to \\
 & & \quad ((\alpha \to \alpha \to \texttt{bool}) \to [\alpha] \to [\alpha]) \cap \\
 & & \quad (\mathit{TotalOrder}.\varepsilon \to \boldsymbol{\omega} \to \mathit{Sorted}(\varepsilon)) \cap \\
 & & \quad (\mathit{PartialOrder}.\eta \to \boldsymbol{\omega} \to \mathit{TopSorted}(\eta))) \\
\quad \ldots & & \\
\quad \texttt{swap} & : & ((\alpha \to \beta \to \gamma) \to (\beta \to \alpha \to \gamma)) \cap \\
 & & (\mathit{Relation}.\varepsilon \to \mathit{Rev}(\mathit{Relation}.\varepsilon)) \\
\quad \leq_{\mathbb{N}} & : & (\texttt{int} \to \texttt{int} \to \texttt{bool}) \cap \mathit{Relation.IncreasingTO} \\
\quad \Phi_{\texttt{Rev}} & : & (\alpha \to \alpha) \cap \\
 & & (\mathit{Rev}(\mathit{Relation.IncreasingTO}) \to \mathit{Relation.DecreasingTO}) \cap \\
 & & (\mathit{Rev}(\mathit{Relation.DecreasingTO}) \to \mathit{Relation.IncreasingTO}) \\
\} & &
\end{array}
$$

Inhabitation goal

$$\Gamma \Vdash ? : [\texttt{int}] \to ([\texttt{int}] \cap \mathit{Sorted}(\mathit{DecreasingTO}))$$

Solution

$$
\begin{array}{l}
\Gamma \Vdash \texttt{S F} \; (\Phi_{\texttt{Rev}} \; (\texttt{swap} \leq_{\mathbb{N}})) : \\
\qquad [\texttt{int}] \to ([\texttt{int}] \cap \mathit{Sorted}(\mathit{DecreasingTO}))
\end{array}
$$

End: Ongoing and Future Work

- Applications
  - Optimizations of **CLS**-algorithm
  - Experiments
  - Staged system
- Further theoretical issues
  - Algorithmic properties of $\leq$
  - Connections to temporal logic synthesis (2-EXPTIME & BCL_0)
  - Process types?