Komponenten- und Service-orientierte Softwarekonstruktion

Vorlesung 6: BCL Type Inhabitation engineered + (CL)S

Boris Düdder
LS XIV – Software Engineering

TU Dortmund
Sommersemester 2015

SS 2015

Relativized Inhabitation

- We consider the *relativized inhabitation* problem:
  - **Given** Γ **and** τ, *does there exist* F *such that* $\Gamma \vdash_{\mathrm{CL}} F : \tau$*?*

Relativized Inhabitation

- We consider the *relativized inhabitation* problem:
  - **Given Γ and τ**, *does there exist F such that $\Gamma \vdash_{\mathrm{CL}} F : \tau$?*
- Relativized inhabitation in simple types is much harder than inhabitation in the fixed theory of $\lambda^{\to}$ (SKI)
  - *Undecidable*: **Linial-Post theorems, 1948 ff.**

Relativized Inhabitation

- We consider the *relativized inhabitation* problem:
  - **Given Γ and τ**, *does there exist F such that $\Gamma \vdash_{\mathrm{CL}} F : \tau$?*
- Relativized inhabitation in simple types is much harder than inhabitation in the fixed theory of $\lambda^{\to}$ (SKI)
  - *Undecidable*: **Linial-Post theorems, 1948 ff.**
- Reason: instead of considering a fixed theory ($\lambda^{\to}$, IPC) we consider an arbitrary input theory

Relativized Inhabitation

- We consider the *relativized inhabitation* problem:
  - **Given Γ and τ**, *does there exist F such that $\Gamma \vdash_{\mathrm{CL}} F : \tau$?*
- Relativized inhabitation in simple types is much harder than inhabitation in the fixed theory of $\lambda^{\to}$ (SKI)
  - *Undecidable*: **Linial-Post theorems, 1948 ff.**
- Reason: instead of considering a fixed theory ($\lambda^{\to}$, IPC) we consider an arbitrary input theory
- *The CLS view:* Already in simple types, relativized inhabitation defines a Turing-complete logic programming language for component composition

Intersection types

Definition 1 (Intersection types)

Let $\mathbb{V}$ denote a denumerable set of *type variables*, ranged over by metavariables $\alpha, \beta, \gamma, \ldots$, and let b range over a set $\mathbb{B}$ of *type constants*. The set $\mathbb{T}_\cap$ of *intersection types*, ranged over by $\tau, \sigma, \rho, \ldots$, is defined inductively by:

- $\alpha \in \mathbb{V} \Rightarrow \alpha \in \mathbb{T}_\cap$
- $b \in \mathbb{B} \Rightarrow b \in \mathbb{T}_\cap$
- $\tau \in \mathbb{T}_\cap, \sigma \in \mathbb{T}_\cap \Rightarrow \tau \to \sigma \in \mathbb{T}_\cap$
- $\tau \in \mathbb{T}_\cap, \sigma \in \mathbb{T}_\cap \Rightarrow \tau \cap \sigma \in \mathbb{T}_\cap$

Intersection types are considered modulo associativity, commutativity and idempotence of intersection: $\tau \cap (\sigma \cap \rho) = (\tau \cap \sigma) \cap \rho, \tau \cap \sigma = \sigma \cap \tau, \tau \cap \tau = \tau$.

Intersection type system $\lambda^{\cap}$

$$\frac{}{\Gamma, x : \tau \vdash x : \tau}(\text{var})$$

$$\frac{\Gamma, x : \tau \vdash M : \sigma}{\Gamma \vdash \lambda x.M : \tau \to \sigma}(\to\text{I})$$

$$\frac{\Gamma \vdash M : \tau \to \sigma \quad \Gamma \vdash N : \tau}{\Gamma \vdash MN : \sigma}(\to\text{E})$$

$$\frac{\Gamma \vdash M : \tau_1 \quad \Gamma \vdash M : \tau_2}{\Gamma \vdash M : \tau_1 \cap \tau_2}(\cap\text{I}) \qquad \frac{\Gamma \vdash M : \tau_1 \cap \tau_2}{\Gamma \vdash M : \tau_i}(\cap\text{E})$$

Major reference for this system (a.k.a. "BCD", Barendregt-Coppo-Dezani):

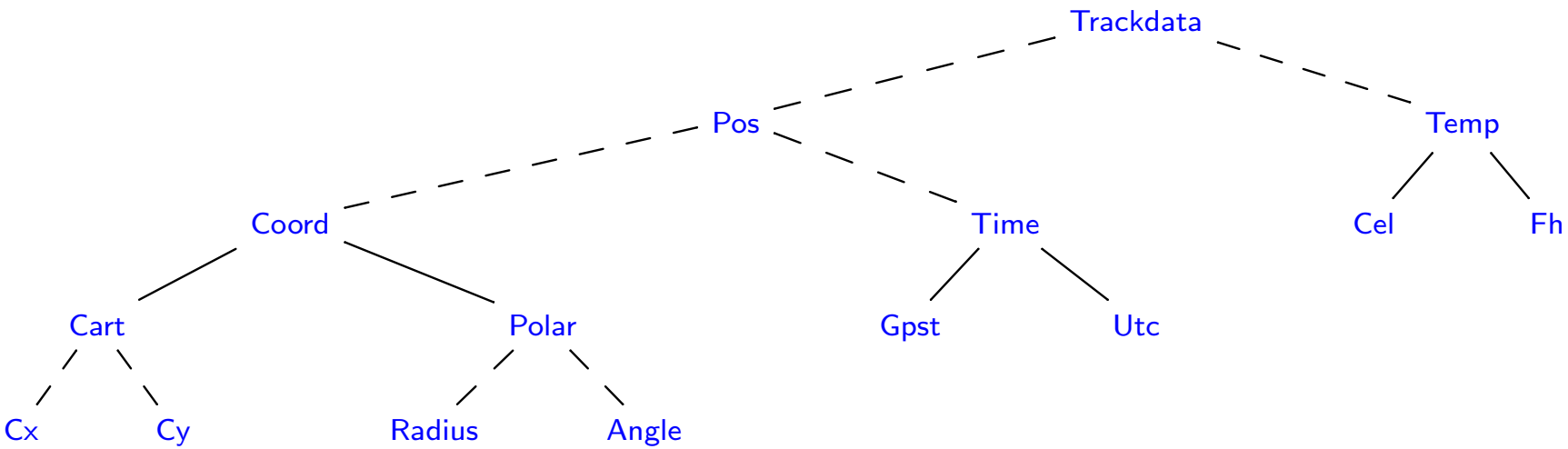
[Barendregt et al., 1983].

Example Repository (Lecture 2)

$$
\Gamma = \{
\begin{array}{lcl}
\texttt{0} & : & \texttt{TrObj} \\
\texttt{Tr} & : & \texttt{TrObj} \to \texttt{D}((\texttt{R},\texttt{R}),\texttt{R},\texttt{R}) \\
\texttt{pos} & : & \texttt{D}((\texttt{R},\texttt{R}),\texttt{R},\texttt{R}) \to ((\texttt{R},\texttt{R}),\texttt{R}) \\
\texttt{cdn} & : & ((\texttt{R},\texttt{R}),\texttt{R}) \to (\texttt{R},\texttt{R}) \\
\texttt{fst} & : & (\texttt{R},\texttt{R}) \to \texttt{R} \\
\texttt{snd} & : & (\texttt{R},\texttt{R}) \to \texttt{R} \\
\texttt{tmp} & : & \texttt{D}((\texttt{R},\texttt{R}),\texttt{R},\texttt{R}) \to \texttt{R} \\
\texttt{cc2pl} & : & ((\texttt{R},\texttt{R}),\texttt{R}) \to ((\texttt{R},\texttt{R}),\texttt{R}) \\
\texttt{cl2fh} & : & \texttt{R} \to \texttt{R}
\end{array}
\}
$$

JR: *Towards Combinatory Logic Synthesis*, BEAT'13

Semantic Type Structure



Semantic Repository

$$
\begin{aligned}
\mathcal{C} = \{ & \\
& \texttt{0} : \texttt{TrObj} \\
& \texttt{Tr} : \texttt{TrObj} \to \texttt{D}((\texttt{R},\texttt{R}) \cap \mathit{Cart}, \texttt{R} \cap \mathit{Gpst}, \texttt{R} \cap \mathit{Cel}) \\
& \texttt{pos} : \texttt{D}((\texttt{R},\texttt{R}) \cap \mathfrak{a}, \texttt{R} \cap \mathfrak{a}', \texttt{R}) \to ((\texttt{R},\texttt{R}) \cap \mathfrak{a}, \texttt{R} \cap \mathfrak{a}') \cap \mathit{Pos} \\
& \texttt{cdn} : ((\texttt{R},\texttt{R}) \cap \mathfrak{a}, \texttt{R}) \cap \mathit{Pos} \to (\texttt{R},\texttt{R}) \cap \mathfrak{a} \\
& \texttt{fst} : ((\texttt{R},\texttt{R}) \cap \mathit{Coord} \to \texttt{R}) \cap \\
& \qquad (\mathit{Cart} \to \mathit{Cx}) \cap (\mathit{Polar} \to \mathit{Radius}) \\
& \texttt{snd} : ((\texttt{R},\texttt{R}) \cap \mathit{Coord} \to \texttt{R}) \cap \\
& \qquad (\mathit{Cart} \to \mathit{Cy}) \cap (\mathit{Polar} \to \mathit{Angle}) \\
& \texttt{tmp} : \texttt{D}((\texttt{R},\texttt{R}), \texttt{R}, \texttt{R} \cap \mathfrak{a}) \to \texttt{R} \cap \mathfrak{a} \\
& \texttt{cc2pl} : (\texttt{R},\texttt{R}) \cap \mathit{Cart} \to (\texttt{R},\texttt{R}) \cap \mathit{Polar} \\
& \texttt{cl2fh} : \texttt{R} \cap \mathit{Cel} \to \texttt{R} \cap \mathit{Fh} \\
\}&
\end{aligned}
$$

Composition Synthesis via Inhabitation

$\mathcal{C} = \{$

$$
\begin{array}{lcl}
\texttt{0} & : & \texttt{TrObj} \\
\texttt{Tr} & : & \texttt{TrObj} \to \texttt{D}((\texttt{R},\texttt{R}) \cap \mathit{Cart}, \texttt{R} \cap \mathit{Gpst}, \texttt{R} \cap \mathit{Cel}) \\
\texttt{pos} & : & \texttt{D}((\texttt{R},\texttt{R}) \cap \mathfrak{a}, \texttt{R} \cap \mathfrak{a}', \texttt{R}) \to ((\texttt{R},\texttt{R}) \cap \mathfrak{a}, \texttt{R} \cap \mathfrak{a}') \cap \mathit{Pos} \\
\texttt{cdn} & : & ((\texttt{R},\texttt{R}) \cap \mathfrak{a}, \texttt{R}) \cap \mathit{Pos} \to (\texttt{R},\texttt{R}) \cap \mathfrak{a} \\
\texttt{fst} & : & ((\texttt{R},\texttt{R}) \cap \mathit{Coord} \to \texttt{R}) \cap \\
 & & (\mathit{Cart} \to \mathit{Cx}) \cap (\mathit{Polar} \to \mathit{Radius}) \\
\texttt{snd} & : & ((\texttt{R},\texttt{R}) \cap \mathit{Coord} \to \texttt{R}) \cap \\
 & & (\mathit{Cart} \to \mathit{Cy}) \cap (\mathit{Polar} \to \mathit{Angle}) \\
\texttt{tmp} & : & \texttt{D}((\texttt{R},\texttt{R}), \texttt{R}, \texttt{R} \cap \mathfrak{a}) \to \texttt{R} \cap \mathfrak{a} \\
\texttt{cc2pl} & : & (\texttt{R},\texttt{R}) \cap \mathit{Cart} \to (\texttt{R},\texttt{R}) \cap \mathit{Polar} \\
\texttt{cl2fh} & : & \texttt{R} \cap \mathit{Cel} \to \texttt{R} \cap \mathit{Fh}
\end{array}
$$

$\}$

Composition Synthesis via Inhabitation

$$
\begin{aligned}
\mathcal{C} = \{ \\
&\texttt{0} && : && \texttt{TrObj} \\
&\texttt{Tr} && : && \texttt{TrObj} \to \texttt{D}((\texttt{R},\texttt{R}) \cap \mathit{Cart}, \texttt{R} \cap \mathit{Gpst}, \texttt{R} \cap \mathit{Cel}) \\
&\texttt{pos} && : && \texttt{D}((\texttt{R},\texttt{R}) \cap \mathfrak{a}, \texttt{R} \cap \mathfrak{a}', \texttt{R}) \to ((\texttt{R},\texttt{R}) \cap \mathfrak{a}, \texttt{R} \cap \mathfrak{a}') \cap \mathit{Pos} \\
&\texttt{cdn} && : && ((\texttt{R},\texttt{R}) \cap \mathfrak{a}, \texttt{R}) \cap \mathit{Pos} \to (\texttt{R},\texttt{R}) \cap \mathfrak{a} \\
&\texttt{fst} && : && ((\texttt{R},\texttt{R}) \cap \mathit{Coord} \to \texttt{R}) \cap \\
& && && (\mathit{Cart} \to \mathit{Cx}) \cap (\mathit{Polar} \to \mathit{Radius}) \\
&\texttt{snd} && : && ((\texttt{R},\texttt{R}) \cap \mathit{Coord} \to \texttt{R}) \cap \\
& && && (\mathit{Cart} \to \mathit{Cy}) \cap (\mathit{Polar} \to \mathit{Angle}) \\
&\texttt{tmp} && : && \texttt{D}((\texttt{R},\texttt{R}), \texttt{R}, \texttt{R} \cap \mathfrak{a}) \to \texttt{R} \cap \mathfrak{a} \\
&\texttt{cc2pl} && : && (\texttt{R},\texttt{R}) \cap \mathit{Cart} \to (\texttt{R},\texttt{R}) \cap \mathit{Polar} \\
&\texttt{cl2fh} && : && \texttt{R} \cap \mathit{Cel} \to \texttt{R} \cap \mathit{Fh} \\
\}
\end{aligned}
$$

$$\mathcal{C} \vdash_{\mathsf{C1}} ? : \texttt{R} \cap \mathit{Fh}$$

Composition Synthesis via Inhabitation

$$
\begin{aligned}
\mathcal{C} = \{ \\
&\texttt{O} && : && \texttt{TrObj} \\
&\texttt{Tr} && : && \texttt{TrObj} \to \texttt{D}((\texttt{R},\texttt{R}) \cap \mathit{Cart}, \texttt{R} \cap \mathit{Gpst}, \texttt{R} \cap \mathit{Cel}) \\
&\texttt{pos} && : && \texttt{D}((\texttt{R},\texttt{R}) \cap \mathfrak{a}, \texttt{R} \cap \mathfrak{a}', \texttt{R}) \to ((\texttt{R},\texttt{R}) \cap \mathfrak{a}, \texttt{R} \cap \mathfrak{a}') \cap \mathit{Pos} \\
&\texttt{cdn} && : && ((\texttt{R},\texttt{R}) \cap \mathfrak{a}, \texttt{R}) \cap \mathit{Pos} \to (\texttt{R},\texttt{R}) \cap \mathfrak{a} \\
&\texttt{fst} && : && ((\texttt{R},\texttt{R}) \cap \mathit{Coord} \to \texttt{R}) \cap \\
& && && (\mathit{Cart} \to \mathit{Cx}) \cap (\mathit{Polar} \to \mathit{Radius}) \\
&\texttt{snd} && : && ((\texttt{R},\texttt{R}) \cap \mathit{Coord} \to \texttt{R}) \cap \\
& && && (\mathit{Cart} \to \mathit{Cy}) \cap (\mathit{Polar} \to \mathit{Angle}) \\
&\texttt{tmp} && : && \texttt{D}((\texttt{R},\texttt{R}), \texttt{R}, \texttt{R} \cap \mathfrak{a}) \to \texttt{R} \cap \mathfrak{a} \\
&\texttt{cc2pl} && : && (\texttt{R},\texttt{R}) \cap \mathit{Cart} \to (\texttt{R},\texttt{R}) \cap \mathit{Polar} \\
&\texttt{cl2fh} && : && \texttt{R} \cap \mathit{Cel} \to \texttt{R} \cap \mathit{Fh} \\
\}
\end{aligned}
$$

$$\mathcal{C} \vdash_{\text{C1}} ? : \texttt{R} \cap \mathit{Fh} \rightsquigarrow \mathcal{C} \vdash_{\text{C1}} \texttt{cl2fh}\ (\texttt{tmp}\ (\texttt{Tr}\ \texttt{O})) : \texttt{R} \cap \mathit{Fh}$$

Composition Synthesis via Inhabitation

$\mathcal{C} = \{$

$$
\begin{array}{lcl}
\texttt{0} & : & \texttt{TrObj} \\
\texttt{Tr} & : & \texttt{TrObj} \to \texttt{D}((\texttt{R},\texttt{R}) \cap \mathit{Cart}, \texttt{R} \cap \mathit{Gpst}, \texttt{R} \cap \mathit{Cel}) \\
\texttt{pos} & : & \texttt{D}((\texttt{R},\texttt{R}) \cap \mathfrak{a}, \texttt{R} \cap \mathfrak{a}', \texttt{R}) \to ((\texttt{R},\texttt{R}) \cap \mathfrak{a}, \texttt{R} \cap \mathfrak{a}') \cap \mathit{Pos} \\
\texttt{cdn} & : & ((\texttt{R},\texttt{R}) \cap \mathfrak{a}, \texttt{R}) \cap \mathit{Pos} \to (\texttt{R},\texttt{R}) \cap \mathfrak{a} \\
\texttt{fst} & : & ((\texttt{R},\texttt{R}) \cap \mathit{Coord} \to \texttt{R}) \cap \\
 & & (\mathit{Cart} \to \mathit{Cx}) \cap (\mathit{Polar} \to \mathit{Radius}) \\
\texttt{snd} & : & ((\texttt{R},\texttt{R}) \cap \mathit{Coord} \to \texttt{R}) \cap \\
 & & (\mathit{Cart} \to \mathit{Cy}) \cap (\mathit{Polar} \to \mathit{Angle}) \\
\texttt{tmp} & : & \texttt{D}((\texttt{R},\texttt{R}), \texttt{R}, \texttt{R} \cap \mathfrak{a}) \to \texttt{R} \cap \mathfrak{a} \\
\texttt{cc2pl} & : & (\texttt{R},\texttt{R}) \cap \mathit{Cart} \to (\texttt{R},\texttt{R}) \cap \mathit{Polar} \\
\texttt{cl2fh} & : & \texttt{R} \cap \mathit{Cel} \to \texttt{R} \cap \mathit{Fh}
\end{array}
$$

$\}$

$$\mathcal{C} \vdash_{\mathsf{C1}} ? : \texttt{R} \cap \mathit{Fh} \rightsquigarrow \mathcal{C} \vdash_{\mathsf{C1}} \texttt{cl2fh}\ (\texttt{tmp}\ (\texttt{Tr}\ \texttt{0})) : \texttt{R} \cap \mathit{Fh}$$

$$\mathcal{C} \vdash_{\mathsf{C1}} ? : \mathit{Radius}$$

Composition Synthesis via Inhabitation

$$
\begin{aligned}
\mathcal{C} = \{ \\
&\texttt{O} && : && \texttt{TrObj} \\
&\texttt{Tr} && : && \texttt{TrObj} \to \texttt{D}((\texttt{R},\texttt{R}) \cap \mathit{Cart}, \texttt{R} \cap \mathit{Gpst}, \texttt{R} \cap \mathit{Cel}) \\
&\texttt{pos} && : && \texttt{D}((\texttt{R},\texttt{R}) \cap \mathfrak{a}, \texttt{R} \cap \mathfrak{a}', \texttt{R}) \to ((\texttt{R},\texttt{R}) \cap \mathfrak{a}, \texttt{R} \cap \mathfrak{a}') \cap \mathit{Pos} \\
&\texttt{cdn} && : && ((\texttt{R},\texttt{R}) \cap \mathfrak{a}, \texttt{R}) \cap \mathit{Pos} \to (\texttt{R},\texttt{R}) \cap \mathfrak{a} \\
&\texttt{fst} && : && ((\texttt{R},\texttt{R}) \cap \mathit{Coord} \to \texttt{R}) \cap \\
& && && (\mathit{Cart} \to \mathit{Cx}) \cap (\mathit{Polar} \to \mathit{Radius}) \\
&\texttt{snd} && : && ((\texttt{R},\texttt{R}) \cap \mathit{Coord} \to \texttt{R}) \cap \\
& && && (\mathit{Cart} \to \mathit{Cy}) \cap (\mathit{Polar} \to \mathit{Angle}) \\
&\texttt{tmp} && : && \texttt{D}((\texttt{R},\texttt{R}), \texttt{R}, \texttt{R} \cap \mathfrak{a}) \to \texttt{R} \cap \mathfrak{a} \\
&\texttt{cc2pl} && : && (\texttt{R},\texttt{R}) \cap \mathit{Cart} \to (\texttt{R},\texttt{R}) \cap \mathit{Polar} \\
&\texttt{cl2fh} && : && \texttt{R} \cap \mathit{Cel} \to \texttt{R} \cap \mathit{Fh} \\
\}
\end{aligned}
$$

$$\mathcal{C} \vdash_{\mathsf{C1}} ? : \texttt{R} \cap \mathit{Fh} \rightsquigarrow \mathcal{C} \vdash_{\mathsf{C1}} \texttt{cl2fh}\ (\texttt{tmp}\ (\texttt{Tr}\ \texttt{O})) : \texttt{R} \cap \mathit{Fh}$$

$$\mathcal{C} \vdash_{\mathsf{C1}} ? : \mathit{Radius} \rightsquigarrow \mathcal{C} \vdash_{\mathsf{C1}} \texttt{fst}\ (\texttt{cdn}\ (\texttt{cc2pl}\ (\texttt{pos}\ (\texttt{Tr}\ \texttt{O})))) : \mathit{Radius}$$

Subtyping

We extend our type system by a subtyping relation and a special type constant ω introduced by Dezani [Barendregt et al., 1983]. The subtyping relation $\leq$ is the least relation satisfying the following axioms:

$$\sigma \leq \omega, \quad \omega \leq \omega \to \omega, \quad \sigma \cap \tau \leq \sigma, \quad \sigma \cap \tau \leq \tau, \quad \sigma \leq \sigma \cap \sigma;$$

$$(\sigma \to \tau) \cap (\sigma \to \rho) \leq \sigma \to \tau \cap \rho;$$

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

We define the symmetric closure of $\leq$ as $=$, e.g. $\sigma = \tau$ iff $\sigma \leq \tau$ and $\tau \leq \sigma$.

Exercise 1

Using the axioms of $\leq$, show that holds:

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

Type Inhabitation Decision Tactic

To answer $\Gamma \vdash ? : \tau$ apply one of the following tactics:

- for $\tau = \tau_1 \to \tau_2$, ask $\Gamma \cup \{\tau_1\} \vdash ? : \tau_2$
- for $\tau = a$, choose $x \in \Gamma$ with $x : \sigma_1 \to \ldots \to \sigma_n \to a$ then ask $\Gamma \vdash ? : \sigma_i$ for all $1 \leq i \leq n$. Success if $n = 0$.

Alternating Turing Machine (ATM

ATM

Tuple $\mathcal{M} = (\Sigma, Q, q_0, q_a, q_r, \Delta)$. Set of states $Q = Q_\exists \uplus Q_\forall$ is partitioned into a

1. set $Q_\exists$ of existential states (CHOOSE) and
2. set $Q_\forall$ of universal states (FORALL).

Initial state $q_0 \in Q$, an accepting state $q_a \in Q_\forall$, and a rejecting state $q_r \in Q_\exists$.

Alternating Turing Machine (ATM

ATM

- An accepting configuration is eventually accepting.
- If $\mathcal{C}$ is existential and some successor of $\mathcal{C}$ is eventually accepting then so is $\mathcal{C}$.
- If $\mathcal{C}$ is universal and all successors of $\mathcal{C}$ are eventually accepting then so is $\mathcal{C}$.

An input is said to be *accepted* by $\mathcal{M}$ if and only if the initial configuration is eventually accepting.

Note: Formally we define the set of all eventually accepting configurations as the smallest set satisfying the appropriate closure conditions.

Levels

Definition 2

(Levels) Given a type τ we define the *level* of τ, written $\ell(\tau)$, as follows.

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

The level of a substitution S, written $\ell(S)$, is defined as

$$\ell(S) = \max\{\ell(S(\alpha)) \mid \alpha \in \mathbb{V}\}.$$

A *level-k type* is a type τ with $\ell(\tau) \leq k$, and a *level-k substitution* is a substitution S with $\ell(S) \leq k$. For $k \geq 0$, we let $\mathbb{T}_k$ denote the set of all level-k types. For a subset A of atomic types, we let $\mathbb{T}_k(A)$ denote the set of level-k types with atoms (leaves) in the set A.

Bounded Combinatory Logic $\mathrm{BCL}_k(\rightarrow, \cap)$ [Düdder et al., 2012]

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

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

Deciding for $\mathrm{BCL}_k(\rightarrow, \cap)$

$Input: \quad \Gamma, \tau, k$

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

Deciding for $\mathrm{BCL}_k(\rightarrow, \cap)$

Input : Γ, τ, k

loop :

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

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

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

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

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

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

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

Deciding for $\mathrm{BCL}_k(\rightarrow, \cap)$

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;

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

Deciding for $\mathrm{BCL}_k(\rightarrow, \cap)$

$\mathit{Input}: \quad \Gamma, \tau, k$

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

$\mathit{loop}:$

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

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

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;

Deciding for $\mathrm{BCL}_k(\rightarrow, \cap)$

$\mathit{Input}: \quad \Gamma, \tau, k$

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

$\mathit{loop}:$

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

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

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;

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

Complexity for Finite and Bounded CL

Theorem 3

[Rehof and Urzyczyn, 2011] For finite combinatory logic FCL*:*

1. *Relativized inhabitation in* $\mathrm{FCL}(\to)$ *is in* PTIME
2. *Relativized inhabitation in* $\mathrm{FCL}(\to, \cap)$ *is* EXPTIME*-complete*

Theorem 4

[Düdder et al., 2012] For bounded combinatory logic BCL_k*:*

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

Optimization Ideas (I)

Speed-up algorithm using logical properties

- term level optimizations
  - normalization of intersection types
  - optimization of subtyping relation $\leq$
  - bounding the substitution in types
  - organization of type environment Γ

Details can be found in [Düdder, 2014].

Optimization Ideas (II)

Speed-up algorithm using efficient computation

- prevention of redundant calculations
  - result caches
  - reusing cached results
  - cycle detection
  - restriction of result sets of inhabitants
- multi-core processing and parallelization
  - rolling processing queues
  - distributed computing barriers

Preventing Failing Subtypes

Redundant calculations are failed inhabitation questions. We can use subtyping to reduce unnecessary inhabitation questions:

Lemma 5

Let τ and τ' be types with $\tau \leq \tau'$.

If $\Gamma \not\vdash ? : \tau'$ holds, then $\Gamma \not\vdash ? : \tau$ also holds.

Exercise 2

Prove Lemma 5.

Path

Definition 6

If $\tau = \tau_1 \to \cdots \to \tau_m \to \sigma$, then we write $\sigma = tgt_m(\tau)$ and $\tau_i = arg_i(\tau)$, for $i \leq m$. We say that σ is a *target* of τ and τ_i is the *i-th argument* of τ.

If $arg_i(\tau) = \rho$ for all i we also write $\tau = \rho^m \to \sigma$. A type of the form $\tau_1 \to \cdots \to \tau_m \to a$, where $a \neq \omega$ is an atom,[1] is called a *path of length m*. A type τ is *organized* if it is a (possibly empty) intersection of paths (those are called *paths in τ*).

[1] Note that $\tau_1 \to \cdots \to \tau_m \to \omega = \omega$.

Path

The path lemma forms the core of the inhabitation algorithm.

Lemma 7

Let $\tau = \bigcap_{i \in I} \tau_i$ *where the* τ_i *are paths and let* $\sigma = \beta_1 \to \ldots \to \beta_n \to p$ *where* $p \neq \omega$ *is an atom.*

We have $\tau \leq \sigma$ *if and only if there is an* $i \in I$ *with* $\tau_i = \alpha_1 \to \ldots \to \alpha_n \to p$ *and* $\beta_j \leq \alpha_j$ *for all* $j \leq n$.

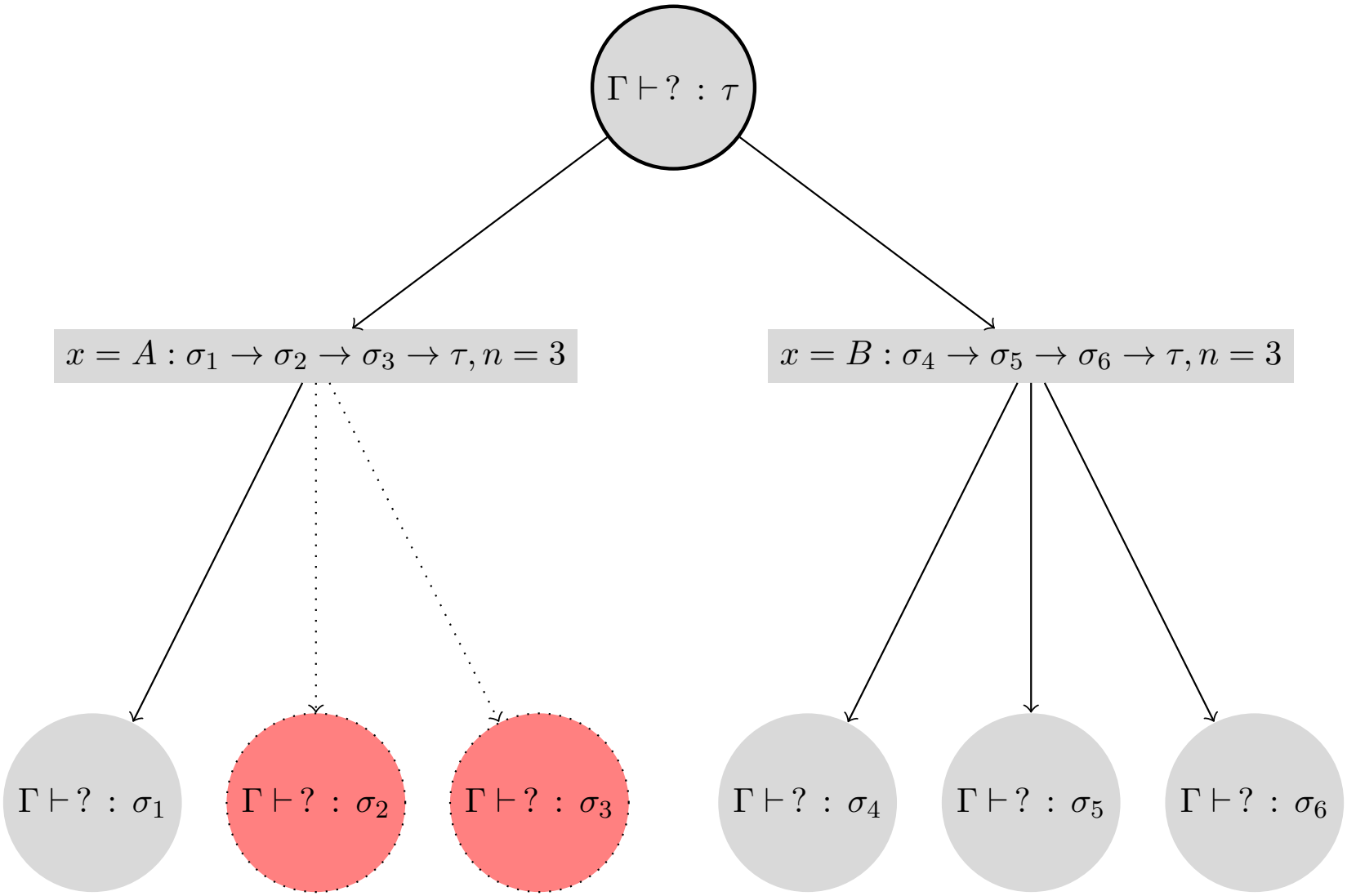
Lookahead Motivation

Assume the following type environment

$$\begin{aligned}\Gamma = \quad \{ & \\ & A : \sigma_1 \to \sigma_2 \to \sigma_3 \to \tau, \\ & B : \sigma_4 \to \sigma_5 \to \sigma_6 \to \tau, \\ & C : \sigma_1, \\ & D : \sigma_4, \\ & E : \sigma_5, \\ & F : \sigma_6 \\ \} & \end{aligned}$$

Combinators with types σ_2 and σ_3 are not present in Γ.

Lookahead Motivation



Lookahead Optimization

$Input:$ Γ, τ— all types in Γ and $\tau = \bigcap_{i \in I} \tau_i$ organized

$loop:$

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

2 write $\sigma \equiv \bigcap_{j \in J} \sigma_j$

3 FOR EACH $i \in I, j \in J, m \leq \|\sigma\|$ DO

4 $\quad$ $\texttt{candidates}(i, j, m) := \texttt{Match}(tgt_m(\sigma_j) \leq \tau_i)$

5 $M := \{m \leq \|\sigma\| \mid \forall_{i \in I} \exists_{j \in J} : \texttt{candidates}(i, j, m) = \textbf{true}\}$

6 CHOOSE $m \in M$;

7 FOR EACH $i \in I$ DO

8 $\quad$ CHOOSE $j_i \in J$ with $\texttt{candidates}(i, j_i, m) = \textbf{true}$

9 $\quad$ CHOOSE S_i a substitution

10 $\quad$ CHOOSE $\pi_i \in \mathbb{P}_m(\overline{S_i(\sigma_{j_i})})$ with $tgt_m(\pi_i) \leq \tau_i$ and

11 $\quad$ $\forall 1 \leq l \leq m \; \forall \pi' \in \overline{arg_l(\pi_i)} \; \exists (y : \rho) \in \Gamma \; \exists$ a path ρ'

12 $\quad$ in $\rho \; \exists k \; : \; \texttt{Match}(tgt_k(\rho') \leq \pi') = \textbf{true}$

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

14 ELSE FORALL$(l = 1 \ldots m)$

15 $\quad\quad$ $\tau := \bigcap_{i \in I} arg_l(\pi_i)$;

16 $\quad\quad$ GOTO $loop$;


□[Distance, R]?
distance
□[Cart, P([R; R])]?
□[Cart, P([R; R])]?
cdn
cdn
cdn
cdn
[Pos, P([[P([R; R]), Cart]; R])]?
[Pos, P([[P([R; R]), Cart]; R]), P([[P([R; R]), Polar]; R])]?
[Pos, P([[P([R; R]), Cart]; R])]?
[Pos, P([[P([R; R]), Cart]; R]), P([[P([R; R]), Polar]; R])]?
pos
pos
D([[Cart, P([R; R])]; [Gpst, R]; R])?
D([[Cart, P([R; R])]; [Gpst, R]; R])?
TrV
TrObj?
O

Optimized Algorithm for BCL Inhabitation I

Require: Γ, $\tau = \bigcap_{i \in I} \tau_i$ are organized

Input: Γ, τ
PossibleSuccess:=**false**
for all $(x : \sigma) \in \Gamma$ **do**
    write $\sigma \equiv \bigcap_{j \in J} \sigma_j$
    for all $i \in I, j \in J, m \leq \|\sigma\|$ **do**
        candidates(i, j, m):=Match$(tgt_m(\sigma_j) \leq \tau_i)$
    end for
    M:=$\{m \leq \|\sigma\| \mid \forall_{i \in I} \exists_{j \in J} : \texttt{candidates}(i, j, m) =$ **true**$\}$
    if M$\neq \emptyset$ **then**
        for all $m \in M$ **do**
            for all $j_i \in J$ with $\texttt{candidates}(i, j_i, m) =$ **true do**
                for all S_i is a substitution **do**
                for all $\pi_i \in \mathbb{P}_m(\overline{S_i(\sigma_{j_i})})$ with $tgt_m(\pi_i) \leq \tau_i$ and
                $\forall 1 \leq l \leq m \; \forall \pi' \in \overline{arg_l(\pi_i)} \; \exists (y : \rho) \in \Gamma \; \exists$ a path ρ' in ρ
                $\exists k \; : \; \texttt{Match}(tgt_k(\rho') \leq \pi') =$ **true do**
                    Add group node to τ
                    if $n = 0$ **then**
                        Mark τ with **true**
                        Propagate result τ in execution graph $\mathcal{G}_E$
                        PossibleSuccess:=**true**

Optimized Algorithm for BCL Inhabitation II

```
19:                     else
20:                         Mark τ with UNKNOWN
21:                         for all l ∈ {1 ... m} do
22:                             Add inhabitation node ⋂_{i∈I} arg_l(π_i) to g
23:                             PossibleSuccess:=true
24:                         end for
25:                     end if
26:                 end for
27:             end for
28:         end for
29:     end for
30: end if
31: end for
32: if PossibleSuccess then
33:     return SUCCESS
34: else
35:     Mark τ as FAILED
36:     return FAILED
37: end if
```

Concurrent Control Algorithm for BCL Inhabitation I

1: *Input:* Γ, τ
2: `Q` := Initialize Working Queue
3: C_S := Initialize Success Cache
4: C_F := Initialize Fail Cache
5: $\overline{\Gamma}$:= Organize Γ
6: τ_N := Normalize τ
7: `root` := Create inhabitation node τ_N
8: Enqueue `root` in `Q`
9: Spawn n working threads executing the task={
10:
11: **while** `Q` has elements **and** (**not** termination signal received) **do**
12: Increase semaphore S
13: q := Dequeue from `Q` (using assignment strategy S)
14: **if** q is part of cycle **then**
15: Mark q as end point of cycle
16: **end if**
17: $\overline{\tau'}$:= Organize τ in q
18: **if** $\overline{\tau'} \in C_S$ **then**
19: Link parent of $\overline{\tau'}$ to existing $\overline{\tau'} \in C_S$
20: **else**
21: **if** $\exists \sigma \in C_F . \overline{\tau'} \leq \sigma$ **then**
22: Mark $\overline{\tau'}$ as *FAILED*

Concurrent Control Algorithm for BCL Inhabitation II

```
23:         end if
24:         if τ̄' ∉ C_F then
25:             if not InhabOptimized(Γ̄,τ̄') then
26:                 Add τ̄' to C_F
27:                 Mark τ̄' as FAILED
28:             end if
29:             Propagate result of τ̄' in execution graph G_E
30:         end if
31:     end if
32:     Decrease semaphore S
33: end while
34: }
35:
36: Wait for all n working threads
37: Fix cycles recursively in root
38: Mark UNKNOWN nodes as FAILED and propagate result
39: if root is marked with SUCCESS then
40:     return ACCEPT
41: else
42:     return FAIL
43: end if
```

Combinatory Logic Synthesizer (CL)S Features

- Theorem prover (proofs-as-programs correspondence)
- Combinatory Logic Synthesis for $\mathrm{BCL}_0(\cap, \leq)$
- Version 1.0
  - Proof-of-concept
  - Enumerates inhabitants (even cyclic ones)
  - Variable kinding
  - Atomic subtyping extension for taxonomies
- Version 2.0
  - Algebraic optimizations
  - Co-variant type constructors

(cf. [Bessai et al., 2014])

Operators and Corresponding Expressions in (CL)S

	Mathematical example	(CL)S representation example
Atoms	τ, σ	`tau`, `sigma`, τ, σ
Variables	α, β	`alpha`, `beta`, α, β
$\rightarrow$	$\tau \rightarrow \sigma$	`tau->sigma` or $\tau \rightarrow \sigma$
$\cap$	$\tau \cap \sigma$	`[tau, sigma]` or `[` τ `,` σ`]`
Covariant constructor	$C(\tau_1, \ldots, \tau_n)$	`C(tau1,...,taun)`
$\leq$	$\tau \leq \sigma$	`tau<=sigma` or $\tau \leq \sigma$
Subst.	$S(\alpha) = \tau$	$\{\alpha\} => \{\tau\}$ $\{\alpha\} \sim> \{\tau\}$

Code Example in (CL)S

Assuming a type environment

$$\Gamma = \{ \\ A : \sigma_1 \to \sigma_2 \cap \sigma_3 \to a, \\ B : \alpha, \\ C : \sigma_2 \cap \sigma_3 \\ \},$$

with the substitution $\{\alpha \mapsto \sigma_1\}$ and the atomic subtyping extension $a \leq a'$ with type atoms a and a'.

Code Example in (CL)S

Then $\Gamma \vdash ? : a'$ can be coded as input for (CL)S as follows:

```
{
   (* Type environment Gamma *)
   A : sigma1 -> [sigma2, sigma3] -> a,
   B : alpha,
   C : [sigma2, sigma3]
},
{
   (* Substitution(s) *)
   {alpha} => {sigma1}
},
{
   (* Atomic subtyping extension *)
   tau<=tau'
}
|- ? : a' (* Inhabitation question *)
```

(CL)S demonstration

Download at `https://depot.tu-dortmund.de/tpqr8`


Barendregt, H., Coppo, M., and Dezani-Ciancaglini, M. (1983).
A Filter Lambda Model and the Completeness of Type Assignment.
Journal of Symbolic Logic, 48(4):931–940.


Bessai, J., Dudenhefner, A., Düdder, B., Martens, M., and Rehof, J. (2014).
Combinatory Logic Synthesizer.
In Margaria, T. and Steffen, B., editors, *ISoLA'14*, volume 8802, pages 26–40.


Düdder, B. (2014).
Automatic Synthesis of Component & Connector-Software Architectures with Bounded Combinatory Logic.
PhD thesis, Technische Universität Dortmund, Fakultät für Informatik, Dortmund, 2014.


Düdder, B., Martens, M., Rehof, J., and Urzyczyn, P. (2012).
Bounded Combinatory Logic.

In *Proceedings of CSL'12*, volume 16 of *LIPIcs*, pages 243–258. Schloss Dagstuhl.

Rehof, J. and Urzyczyn, P. (2011).
Finite Combinatory Logic with Intersection Types.
In *Proceedings of TLCA'11*, volume 6690 of *LNCS*, pages 169–183. Springer.