Coinductive Uniform Proofs

Presented by
Mr Yue Li

Researched by
Dr Henning Basold
Dr Ekaterina Komendantskaya
Mr Yue Li

1 Heriot-Watt University, Scotland
2 ENS de Lyon, France

8 July 2018, Oxford, UK
**Introduction**

**Context** First-order Horn clause logic programming.

**Goal** Detecting non-termination by coinductive proof.

**State of the Art** Heuristic algorithms

1. Coinductive Logic Programming
2. Proof Relevant Corecursive Resolution

**Open Problems** The heuristic algorithms:

1. have limits, and
2. do not have proof theoretic foundation.

**We Propose: Coinductive Uniform Proof**

- A principled approach to the Goal.
- A proof theoretic foundation for the heuristic algorithms.
- Breaking through the limits of the heuristic algorithms.
Background: Fixed-point Models (aka Herbrand Models)

- Given a first-order Horn clause logic program $P$, in classical logic:
- The least fixed-point model contains all finite terms that can be proved to be true w.r.t $P$.
- The greatest fixed-point model contains all finite and infinite terms that cannot be proved to be false (i.e. either true or non-terminating) w.r.t $P$.

Example

- Clauses $\text{nat } 0$ and $\forall x. \text{nat } x \rightarrow \text{nat } (s \ x)$ intend to define the set $\mathbb{N}$ of all non-negative integers.
- A typical $n \in \mathbb{N}$ has the form $s - \cdots - s - 0$.
- The least fixed point model is $M_\mu = \{ \text{nat } n \mid n \in \mathbb{N} \}$.
- The greatest fixed-point model is $M_\nu = M_\mu \cup \{ \text{nat } \omega \}$
- $\ldots$ where $\omega$ is the infinite term $s - s - s - \cdots$. 
Background: Coinductive Logic Programming (CoLP)

- Created by Gopal Gupta et al in 2006
- A goal succeeds if it unifies a previous goal (no occurs check)
- Being sound w.r.t. the greatest fixed-point model.

Example

- Consider the program: $\forall x. \text{zeros } x \rightarrow \text{zeros } [0 \mid x]$
- SLD-derivation ($\leadsto$): $\text{zeros } x \leadsto \text{zeros } x' \leadsto \cdots$

  Result $[0 \mid x']/x, [0 \mid x'']/x', \cdots$
  - leading towards the correct answer, but
  - non-terminating

- CoLP derivation ($\leadsto$): $\text{zeros } x \leadsto \text{zeros } x' \checkmark$

  zeros $x'$ unifies $\text{zeros } x$.

  Result $[0 \mid x]/x$ (circular unifier, representing $[0, 0, \cdots]/x$)
  - giving exactly the correct answer.
Background: Proof Relevant Corecursive Resolution (Precor)

- Created by Komendantskaya et al in 2015
- Including a heuristic to suggest a “coinductive invariant (Co-I)”, plus a specially suggested calculus to prove the Co-I.
- The corresponding infinite SLD-derivation is recoverable from a Precor proof.
Background: Proof Relevant Corecursive Resolution (Precor)

Example

Consider the program: \( \forall x. \text{paul\_loves (dog\_of } x) \rightarrow \text{paul\_loves } x \)
Background: Proof Relevant Corecursive Resolution (Precor)

Example

Consider the program: \( \forall x. p (d x) \rightarrow p x \)
Example

- Consider the program: $\forall x. \ p (d \ x) \rightarrow p \ x$
- SLD-derivation ($\rightsquigarrow$): $p \ x \rightsquigarrow p (d \ x) \rightsquigarrow \cdots$

  **Note** SLD-derivation is restricted to rewriting.
  - non-terminating, no answer.
- CoLP-derivation ($\rightsquigarrow$): $p \ x \rightsquigarrow p (d \ x)$ ✓
  - $p \ x$ unifies $p \ (d \ x)$.

**Result** $(d \ x)/x$ (circular unifier, denoting $[d \rightarrow d \rightarrow \cdots /x]$)
  - A correct answer!

- Precor suggests a Co-I: $\forall x. \ p \ x$
  - then proves the Co-I: $\forall x. \ p \ x \rightarrow p \ c \rightsquigarrow p \ (d \ c)$ ✓
  - $\rightarrow$ is introduction rule for $\forall$; $p \ (d \ c)$ is an instance of the Co-I.

  - The Co-I is a correct and more general (than CoLP) answer.
  - The pattern of the SLD-derivation is captured.
Background: Limitations of CoLP and Precor

- CoLP only works with cyclic patterns.
- Precor requires that SLD-resolution is restricted to term matching (rewriting).

Motivating Example

\[ \forall xy. \text{from } (s \ x) \ y \rightarrow \text{from } x \ [x \mid y] \]

- The “from” predicate has two arguments:
- The first argument takes some number \( N \).
- The second argument returns a stream led by \( N \):

\[ N, s \ N, s \ (s \ N), s \ (s \ (s \ N)), \ldots \]
Background: Limitations of CoLP and Precor

Motivating Example

\[ \forall xy. \text{from } (s \ x) \ y \rightarrow \text{from } x \ [x \mid y] \]

- Task I: Find \( x \) and \( y \), such that, from \( x \ y \)

Approach: SLD

- SLD-derivation (\( \rightsquigarrow \)): from \( x \ y \) \( \rightsquigarrow \) from \( (s \ x) \ y' \) \( \rightsquigarrow \) \( \cdots \)

Result \( [x \mid y']/y, [(s \ x) \mid y'']/y', \cdots \)

Note Full SLD-resolution is needed, instead of just rewriting.

Note Goals do unify (no occurs check)

- leading towards the correct answer only for \( y \)
- non-terminating, no answer for \( x \)
Background: Limitations of CoLP and Precor

Motivating Example

\[ \forall xy. \text{from } (s \, x) \, y \rightarrow \text{from } x \, [x \mid y] \]

▶ Task I: Find \( x \) and \( y \), such that, from \( x \, y \)

Approach: CoLP

▶ CoLP-derivation (\( \rightsquigarrow \)): from \( x \, y \) \( \rightsquigarrow \) from \( (s \, x) \, y' \)

▶ from \( x \, y \) unifies from \( (s \, x) \, y' \)

Result \[ [s \, x)/x, [x \mid y]/y \]

😊 A correct pair of answers for both \( x \) and \( y \)!
Background: Limitations of CoLP and Precor

Motivating Example

∀xy. from (s x) y → from x [x | y]

- Task I: Find x and y, such that, from x y

Approach: Precor

- N/A
  - because full SLD-resolution is needed, instead of just rewriting.
Background: Limitations of CoLP and Precor

Motivating Example

\[ \forall x y. \text{from } (s \ x) \ y \rightarrow \text{from } x \ [x \mid y] \]

- Task II: Find \( y \), such that, from 0 \( y \)

Approach: SLD

- SLD-derivation (\( \rightsquigarrow \)): from 0 \( y \) \( \rightsquigarrow \) from (s 0) \( y' \) \( \rightsquigarrow \) \( \cdots \)

Result

\[ [0 \mid y']/y, [(s 0) \mid y'']/y', \cdots \]

Note

Full SLD-resolution is needed, instead of just rewriting.

Note

Goals do not unify (no occurs check)

- leading towards the correct answer, but
- non-terminating
Background: Limitations of CoLP and Precor

Motivating Example

\[ \forall xy. \text{from} (s \ x) \ y \rightarrow \text{from} \ x \ [x \mid y] \]

- Task II: Find \( y \), such that, from \( 0 \ y \)

Approach: CoLP

- CoLP-derivation (\( \rightsquigarrow \)): from \( 0 \ y \) \( \rightsquigarrow \) \( \ldots \)
- CoLP behaves the same as SLD in this case,
- because goals do not unify (no occurs check).

😊 leading towards the correct answer, but

😢 non-terminating
Background: Limitations of CoLP and Precor

Motivating Example

\[ \forall xy. \text{from } (s \times) y \rightarrow \text{from } x \ [x \mid y] \]

- Task II: Find \( y \), such that, from 0 \( y \)

Approach: Precor

- N/A
  - because full SLD-resolution is needed, instead of just rewriting.
Background: Limitations of CoLP and Precor

Motivating Example

\[ \forall xy. \text{from } (s \ x) \ y \rightarrow \text{from } x \ [x \mid y] \]

- Task I: Find \( x \) and \( y \), such that, from \( x \ y \)
- Task II: Find \( y \), such that, from \( 0 \ y \)

Approaches: Summary

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Task I</th>
<th>Task II</th>
</tr>
</thead>
<tbody>
<tr>
<td>SLD</td>
<td>😞</td>
<td>😞</td>
</tr>
<tr>
<td>CoLP</td>
<td>😊</td>
<td>😞</td>
</tr>
<tr>
<td>Precor</td>
<td>😞</td>
<td>😞</td>
</tr>
</tbody>
</table>

😊 At least one answer. 😞 No answer.
Coinductive Uniform Proof (CUP): Motivation

Motivating Example

\[ \forall xy. \text{from} (s \ x) y \rightarrow \text{from} \ x \ [x \mid y] \]

Approach: CUP

▶ We need a representation for the stream.
▶ Let \( f \ N \) denote the stream: \( N, s \ N, s \ (s \ N), s \ (s \ (s \ N)), \cdots \)
▶ Later we will give \( f \) as a (higher-order) \( \lambda \)-term.
▶ Then \( f \ (s \ N) \) denotes the stream: \( s \ N, s \ (s \ N), s \ (s \ (s \ N)), \cdots \)
▶ So we have \( f \ N \equiv [ N \mid f \ (s \ N) ] \)
▶ where \( \equiv \) denotes equality.
Coinductive Uniform Proof (CUP): Motivation

Motivating Example

\( \forall xy. \text{from} (s \, x) \, y \rightarrow \text{from} \, x \, [x \mid y] \)

Approach: CUP

- \( \ldots \) we have \( f \, N \equiv [N \mid f \, (s \, N)] \)
- Let Co-l be: \( \forall x. \text{from} \, x \, (f \, x) \)
- CUP (sketch):
  - **Step 1** \( \forall x. \text{from} \, x \, (f \, x) \rightarrow \text{from} \, c \, (f \, c) \)
  - **Step 2** \( \text{from} \, c \, (f \, c) \equiv \text{from} \, c \, [c \mid f \, (s \, c)] \)
  - **Step 3** \( \text{from} \, c \, [c \mid f \, (s \, c)] \leadsto \text{from} \, (s \, c) \, (f \, (s \, c)) \checkmark \)

- \( \text{from} \, (s \, c) \, (f \, (s \, c)) \) is an instance of Co-l, with substitution \( [s \, c/x] \).
Coinductive Uniform Proof (CUP): Motivation

Motivating Example

\[ \forall xy. \text{from } (s \ x) \ y \rightarrow \text{from } x \ [x \mid y] \]

Approach: CUP

- Using the CUP proof, we can recover the SLD-derivation for an arbitrary instance from \( t (f t) \) of the Co-I: \( \forall x. \text{from } x (f x) \).
  1. The substitution is \( [t/x] \) when we get the instance from the Co-I.
  - Recall the CUP proof: \( \forall x. \text{from } x (f x) \rightarrow \text{from } c (f c) \equiv \text{from } c [c \mid f (s c)] \leadsto \text{from } (s c) (f (s c)) \)
  - We need the segment \( \kappa : \text{from } c (f c) \rightarrow \text{from } (s c) (f (s c)) \)
  2. The substitution is \( [s c/x] \) when we apply Co-I to terminate the proof.
  3. Using substitutions \( [t/x] \) and \( [s c/x] \), we can generate an infinite set \( \Theta \) of substitutions \( [t/c, s t/c, s(s t)/c, s(s(s t))/c, \ldots] \)
  4. We assemble all members of \( \{ \kappa \sigma \mid \sigma \in \Theta \} \) to get:
    - \( \text{from } t (f t) \rightarrow \text{from } (s t) (f (s t)) \rightarrow \text{from } (s(s t)) (f (s(s t))) \cdots \)
  5. \( \ldots \) which is just the SLD-derivation (replacing \( \rightarrow \) by \( \leadsto \)
Coinductive Uniform Proof (CUP): Motivation

Motivating Example

\[ \forall xy. \text{from } (s x) y \rightarrow \text{from } x [x \mid y] \]

Approach: CUP

- The pattern of SLD-derivation is captured by the CUP proof.
- The Co-I (\(\forall x. \text{from } x (f x)\)) is a more general answer than that (\(\text{from } x y \text{ where } [(s x)/x, [x \mid y]/y]\)) given by CoLP.
To represent \( f \), we need fixed-point terms.

To prove universally quantified Co-I, we need hereditary Harrop formula and uniform proof.

To apply the Co-I in later stage of the proof, we need a coinductive proof principle.

To prevent unsound application of Co-I, we need a guarding mechanism.

The system is sound w.r.t

1. The greatest fixed-point model
2. Intuitionistic sequent calculus extended with later modality.
Overview of Term Syntax

The Set $\Lambda_\Sigma$ of Well Formed Terms on $\Sigma$

- Simply typed $\lambda$-terms extended with the $\text{fix}$ binder to denote fixed-points.

\[ \frac{\Sigma; \Gamma, x : \tau \vdash M : \tau}{\Sigma; \Gamma \vdash \text{fix} \, x. \, M : \tau} \quad \text{compare with:} \quad \frac{\Sigma; \Gamma, x : \sigma \vdash M : \tau}{\Sigma; \Gamma \vdash \lambda x. \, M : \sigma \rightarrow \tau} \]

- $\text{fix} \, x. \, M$ is supposed to be equal to $M [\text{fix} \, x. \, M/x]$.

The Set $\Lambda^G_\Sigma$ of Guarded Well Formed Terms

- Guarded terms are particular well formed terms.
- A guarded term models either a finite or an infinite term that occurs in first-order Horn clause logic programming.
Low level details ahead
The Type System

**Definition**

- $\mathbb{B} \rightarrow$ The set of *base type*. $o \notin \mathbb{B} = \{\iota\}$.
- $\mathbb{T} \rightarrow$ The set of *(simple)* types. $\tau \in \mathbb{T} ::= \mathbb{B} \mid \mathbb{B} \rightarrow \mathbb{T}$
- $\mathbb{P} \rightarrow$ The set of *proposition types*. $\rho \in \mathbb{P} ::= o \mid \mathbb{B} \rightarrow \mathbb{P}$

- We adopt the usual convention that $\rightarrow$ binds to the right.

**Order**

$\text{ord}(\iota) = \text{ord}(o) = 0$; all other types $\pi \in \mathbb{T} \cup \mathbb{P}$ have $\text{ord}(\pi) = 1$.

**Arity**

$\text{ar}(\iota) = \text{ar}(o) = 0$; if $\pi \in \mathbb{T} \cup \mathbb{P}$ then $\text{ar}(\iota \rightarrow \pi) = \text{ar}(\pi) + 1$.

**Example**

- $\mathbb{T} = \{\iota, \iota \rightarrow \iota, \iota \rightarrow \iota \rightarrow \iota, \ldots\}$. $\mathbb{P} = \{o, \iota \rightarrow o, \iota \rightarrow \iota \rightarrow o, \ldots\}$.
- In other words, any $\tau \in \mathbb{T}$ can be depicted as $\iota^{\text{ar}(\tau)} \rightarrow \iota$, any $\rho \in \mathbb{P}$ can be depicted as $\iota^{\text{ar}(\rho)} \rightarrow o$. 

Signature and Context

Definition

**Con** — A countable set of constants \(a, b, c, \ldots\).

**Var** — A countable set of variables \(x, y, z, \ldots\).

**\(\Sigma\)** — A *signature*, \(\text{Con} \mapsto (\mathbb{T} \cup \mathbb{P})\).

- **\(\Sigma_T\)** — The set of *term symbols* in \(\Sigma\) with types in \(\mathbb{T}\).
  - \(\Sigma^n_T\) is the subset \(\{c : \tau \in \Sigma_T \mid \text{ord}(\tau) \leq n\}\) of \(\Sigma_T\).

- **\(\Sigma_P\)** — The set of *predicate symbols* in \(\Sigma\) with types in \(\mathbb{P}\).
  - \(\Sigma^n_P\) is the subset \(\{r : \rho \in \Sigma_P \mid \text{ord}(\rho) \leq n\}\) of \(\Sigma_P\).

**\(\Gamma\)** — A *context*, \(\text{Var} \mapsto \mathbb{T}\).

- **\(\Gamma_T\)** — A synonym of \(\Gamma\).
  - \(\Gamma^n_T\) is the subset \(\{x : \tau \in \Gamma_T \mid \text{ord}(\tau) \leq n\}\) of \(\Gamma_T\).

Example

Let \(\Sigma = \{a : \iota\}\) then \(\Sigma_T = \Sigma^1_T = \Sigma^0_T \ni a\)

Let \(\Gamma = \{y : \iota \rightarrow \iota\}\) then \(\Gamma_T = \Gamma^1_T \ni y \notin \Gamma^0_T = \emptyset\)
The Set $\Lambda_\Sigma$ of Well Formed Terms on $\Sigma$

**Definition**

$M \in \Lambda_\Sigma$ iff $\Sigma; \Gamma \vdash_{(m;n)} M : \tau$ for some order constraints $m, n \geq 0$, and $\tau \in \mathbb{T}$. We write $\Sigma; \Gamma \vdash^*_{(m;n)} M : \tau$ only if $\Sigma; \Gamma \vdash_{(m;n)} M : \tau$ and $M$ does not contain any of $\{\text{fix, } \lambda\}$.

\[
\begin{array}{c}
c : \tau \in \Sigma^m_T \\
\hline
\Sigma; \Gamma \vdash_{(m;n)} c : \tau
\end{array} \quad \begin{array}{c}
x : \tau \in \Gamma^n_T \\
\hline
\Sigma; \Gamma \vdash_{(m;n)} x : \tau
\end{array}
\]

\[
\begin{array}{c}
\Sigma; \Gamma \vdash_{(m;n)} M : \sigma \rightarrow \tau \\
\hline
\Sigma; \Gamma \vdash_{(m;n)} M N : \tau
\end{array} \quad \begin{array}{c}
\Sigma; \Gamma \vdash_{(m;n)} N : \sigma \\
\hline
\Sigma; \Gamma \vdash_{(m;n)} M N : \tau
\end{array}
\]

\[
\begin{array}{c}
\Sigma; \Gamma, x : \sigma \vdash_{(m;n)} M : \tau \\
\hline
\Sigma; \Gamma \vdash_{(m;n)} \lambda x. M : \sigma \rightarrow \tau
\end{array} \quad \begin{array}{c}
\Sigma; \Gamma, x : \tau \vdash_{(m;n)} M : \tau \\
\hline
\Sigma; \Gamma \vdash_{(m;n)} \text{fix} x. M : \tau
\end{array}
\]

**Figure:** Definition of $\Sigma; \Gamma \vdash_{(m;n)} M : \tau$. 
The Set $\Lambda_\Sigma$ of Well Formed Terms on $\Sigma$

**Example**

- Let $\Sigma = \{a : \iota, f : \iota \to \iota\}$, $\Gamma = \{y : \iota \to \iota\}$.
- **Provable:** $\Sigma; \Gamma \vdash_{(1;1)} y \ a : \iota$
- **Not provable:** $\Sigma; \Gamma \not\vdash_{(1;0)} y \ a : \iota$

↑ Mind the order constraints.

- **Provable:**
  \[
  \begin{align*}
  \Sigma; \emptyset &\vdash_{(1;0)} \lambda x. \ f \ x : \iota \to \iota \\
  \Sigma; \emptyset &\vdash_{(1;0)} \text{fix} \ x. \ f \ x : \iota
  \end{align*}
  \]

- **Not provable:**
  \[
  \begin{align*}
  \Sigma; \emptyset &\vdash^\ast_{(1;0)} \lambda x. \ f \ x : \iota \to \iota \\
  \Sigma; \emptyset &\vdash^\ast_{(1;0)} \text{fix} \ x. \ f \ x : \iota
  \end{align*}
  \]

↑ Mind the $^\ast$, and note that $\lambda x. \ f \ x$ and $\text{fix} \ x. \ f \ x$ contain the binders $\text{fix}, \lambda$. 


The Set $\Lambda^G_\Sigma$ of Guarded Well Formed Terms

Definition

If $\Sigma; \emptyset \vdash M : \tau$ then $M$ is a guarded fixed-point. If $\Sigma; \Gamma \vdash_M M : \nu$, then $M$ is a guarded well formed term. We denote the set of all guarded well formed terms on $\Sigma$ by $\Lambda^G_\Sigma$.

\[
\begin{align*}
\Sigma; \Gamma \vdash^{*}_{(1;0)} M : \nu & \quad \frac{\Sigma; \emptyset \vdash M : \tau \quad \text{ar}(\tau) = |\vec{N}| \quad \{\Sigma; \Gamma \vdash^{*}_{(1;0)} N : \nu \mid N \in \vec{N}\}}{\Sigma; \Gamma \vdash_M M \vec{N} : \nu} \\
\{\Sigma; \vec{x} : \nu \vdash^{*}_{(1;0)} N : \nu \mid N \in \vec{N}_{1(2,3)}\} & \quad \left[\begin{array}{l}
f : \tau' \in \Sigma^1_\tau \quad \text{ar}(\tau') = |\vec{N}_1| + 1 + |\vec{N}_3| \\
y : \tau \notin \vec{x} \quad \text{ar}(\tau) = |\vec{x}| = |\vec{N}_2|
\end{array}\right] \\
\Sigma; \emptyset \vdash \text{fix } y. \lambda \vec{x}. f \vec{N}_1 (y \vec{N}_2) \vec{N}_3 : \tau
\end{align*}
\]

Figure: Definition of $\Sigma; \Gamma \vdash_M M : \tau$ and $\Sigma; \Gamma \vdash_M M : \tau$
The Set $\Lambda^G_\Sigma$ of Guarded Well Formed Terms

Example

- Recall: we let $f z$ denote the stream:
  
  $z, s z, s (s z), s (s (s z)), \cdots$

- Now we give $f$ as $\text{fix } y. \lambda x. [x \mid y (s x)]$.

- We justify this definition later using the notion of reductions.

- Let $\Sigma = \{[\_ \mid \_] : \iota \rightarrow \iota \rightarrow \iota, s : \iota \rightarrow \iota\}$, we have

  $\Sigma; \emptyset \vdash_{\triangleright} f : \iota \rightarrow \iota$

  and

  $\Sigma; z : \iota \vdash_{g} f \ z : \iota$
The Set $\Lambda^G_\Sigma$ of Guarded Well Formed Terms

Note that:

- By Def. of $\Lambda^G_\Sigma$, there is at most one variable $y : \tau$ bound by fix within any given $M \in \Lambda^G_\Sigma$.
- By Def. of $\mathbb{T} \ni \tau$, $\text{ord}(\tau)$ can only be 0 or 1.

Definition

- $M \in \Lambda^G_\Sigma$ is **first-order** if either 1) $M$ does not contain fix, or 2) there exist $y : \tau$ fix-bound in $M$ and $\text{ord}(\tau) = 0$.
- $M \in \Lambda^G_\Sigma$ is **higher-order** if there exist $y : \tau$ fix-bound in $M$ and $\text{ord}(\tau) = 1$.

Example

- $f z$, i.e. $\text{fix } y : \iota \to \iota. \lambda x. [x | y (s \ x)] z$ is higher-order.
- $\text{fix } y : \iota. [0 | y]$ is first-order.
Well Formed Formulae

Definition

$\varphi$ is a atomic formula on $\Sigma$ if $\Sigma; \Gamma \vdash a \varphi$ for some $\Gamma$; $\varphi$ is a well formed formula (wff) on $\Sigma$ if $\Sigma; \Gamma \vdash \varphi$ for some $\Gamma$. A wff $\varphi$ is closed if $\Sigma; \emptyset \vdash \varphi$.

\[
\begin{array}{c}
(p : \iota^n \rightarrow o) \in \Sigma^1_{\Pi} \quad \{\Sigma; \Gamma \vdash g \ M_k : \iota \mid 1 \leq k \leq n\}
\
\Sigma; \Gamma \vdash a \ p \ M_1 \cdots \ M_n
\
\Sigma; \Gamma \vdash \varphi
\end{array}
\]

\[
\begin{array}{c}
\Sigma; \Gamma \vdash \varphi
\quad \Gamma, x : \iota \vdash \varphi
\quad \Gamma, x : \iota \vdash \varphi
\quad \Sigma; \Gamma \vdash \forall x : \iota. \varphi
\quad \Sigma; \Gamma \vdash \exists x : \iota. \varphi
\end{array}
\]

\[
\begin{array}{c}
\Sigma; \Gamma \vdash \varphi
\quad \Sigma; \Gamma \vdash \psi
\quad \Box \in \{\land, \lor, \rightarrow\}
\end{array}
\]

\[
\Sigma; \Gamma \vdash \varphi \Box \psi
\]

Figure: Formulae
Well Formed Formulae

Definition
A well formed formula $\varphi$ is first-order if all terms involved are first-order. Otherwise $\varphi$ is higher-order.

Example

- $\forall \vec{x} : \iota. \text{from } (s \ x_1) \ x_2 \rightarrow \text{from } x_1 \ [x_1 \mid x_2]$ is first-order (and closed).

- $\forall x : \iota. \text{from } x \ (f \ x)$, where $f$ is $\text{fix } y : \iota \rightarrow \iota. \lambda z. [z \mid y \ (s \ z)]$, is higher-order (and closed).
Hereditary Harrop Formula for Coinductive Uniform Proof

\( A \) — The set of atomic formulae on \( \Sigma \).

\( G \) — The set of well formed hereditary Harrop goal formulae.

\[
G ::= A \mid G \land G \mid G \lor G \mid \exists x : \iota. G \mid D \rightarrow G \mid \forall x : \iota. G
\]

\( D \) — The set of well formed hereditary Harrop program formulae.

\[
D ::= A \mid G \rightarrow D \mid D \land D \mid \forall x : \iota. D
\]

\((G', D')\) The pair of subsets of \( G \) and \( D \) containing all and only closed formulae.

- We take \((G', D')\) as the abstract language for coinductive uniform proof.
Hereditary Harrop Formula for Coinductive Uniform Proof

Definition

- A program is a subset of $D'$.
- A goal is a member of $G'$.

Example

The two formulae below consist of a program:

1. $\forall \vec{x} : \iota. \text{from } (s \, x_1) \, x_2 \rightarrow \text{from } x_1 \, [x_1 \mid x_2]$
2. $\forall x : \iota. \text{from } x \, (f \, x)$

Either formula above can be a goal.
Equivalence Relation for Terms and Formulae

Definition
On terms in $\Lambda_\Sigma$:

- **$\beta$-reduction** ($\rightarrow_\beta$): $(\lambda x. M)N \rightarrow_\beta M[N/x]$
- **fix-reduction** ($\rightarrow_{\text{fix}}$): (fix $x$. $M$) $\rightarrow_{\text{fix}} M[f\text{ix } x. M/x]$
- **combined reduction** ($\rightarrow$): The union of the compatible closures (reductions under applications and binders) of $\rightarrow_\beta$ and $\rightarrow_{\text{fix}}$.
- **convertible relation** ($\equiv$): The equivalence closure of $\rightarrow$.
- **convertible atoms**: Two atoms $p M_1 \cdots M_n \equiv p M'_1 \cdots M'_n$ if $M_k \equiv M'_k$ for $k = 1, \ldots, n$. 
Equivalence Relation for Terms and Formulae

Example
We use $f$ to abbreviate $\text{fix } y. \lambda x. [x \mid y (s x)]$. The following terms are convertible ($\equiv$).

- $f z$
- $[z \mid f (s z)]$
- $[z, s z \mid f (s (s z))]$
- $[z, s z, s (s z) \mid f (s (s (s z))))$
- $\ldots$

This justifies our representation of the stream $z, s z, s (s z), s (s (s z)), \ldots$ by $f z$. 
Coinductive Proof Principle

- \( \Sigma; P; \Delta \rightarrow \varphi \) means \( \varphi \) has a uniform proof w.r.t program \( P \cup \Delta \) on \( \Sigma \).

- \( \Sigma; P \leftrightarrow \varphi \) means \( \varphi \) is coinductively provable from program \( P \) on \( \Sigma \). \( \varphi \) is called a coinductive invariant.

- The rule for \( \Sigma; P \leftrightarrow \varphi \) is:

\[
\begin{align*}
\Sigma; P; \varphi \rightarrow \langle \varphi \rangle \\
\Sigma; P \leftrightarrow \varphi & \quad \text{CO-FIX}
\end{align*}
\]

where \( \varphi \in M' \), \( \langle \rangle \) regulates the proof of \( \Sigma; P; \varphi \rightarrow \varphi \).

Reads If \( \Sigma; P; \varphi \rightarrow \varphi \) in a regulated way, then \( \Sigma; P \leftrightarrow \varphi \).

\( M \) — The intersection of \( D \) and \( G \), given as

\[
M ::= A \mid M \land M \mid M \rightarrow M \mid \forall x : \iota. M
\]

\( M' \) is the subset of \( M \) containing all and only closed formulae.
Uniform Proof

- Developed by Dale Miller et al in 1990s
- The top-level logical constant in a goal determines the goal(s) to prove next.
- A proof theoretic foundation for logic programming
- A criterion to judge logic programming languages.
  - A language $L$ is suitable for logic programming, if the proposition below is true.
  - There is a uniform proof in $L$ iff there is an intuitionistic proof in $L$.
- Four languages satisfy this criterion: first/higher-order Horn clause/hereditary Harrop formula
- No fixed-point terms. More complex type system.
### Uniform Proof

<table>
<thead>
<tr>
<th>Rule</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Sigma; P; \Delta \rightarrow A$</td>
<td>$D \in P \cup \Delta$</td>
</tr>
<tr>
<td>$\Sigma; P; \Delta \rightarrow A$</td>
<td>DECIDE</td>
</tr>
<tr>
<td>$\Sigma; P; \Delta \rightarrow A$</td>
<td>$A \equiv A'$</td>
</tr>
<tr>
<td>$\Sigma; P; \Delta \rightarrow A$</td>
<td>INITIAL</td>
</tr>
<tr>
<td>$\Sigma; P; \Delta \rightarrow A$</td>
<td>$\Sigma; P; D; \Delta \rightarrow G$</td>
</tr>
<tr>
<td>$\Sigma; P; \Delta \rightarrow G$</td>
<td>$\rightarrow L$</td>
</tr>
<tr>
<td>$\Sigma; P; \Delta \rightarrow G$</td>
<td>$\rightarrow R$</td>
</tr>
<tr>
<td>$\Sigma; P; \Delta \rightarrow G$</td>
<td>$\Sigma; P; \Delta \rightarrow \forall x : \iota. , G$</td>
</tr>
<tr>
<td>$\Sigma; P; \Delta \rightarrow G$</td>
<td>$\forall R$</td>
</tr>
<tr>
<td>$\Sigma; P; \Delta \rightarrow G$</td>
<td>$\exists x : \iota. , G$</td>
</tr>
<tr>
<td>$\Sigma; P; \Delta \rightarrow G$</td>
<td>$\exists R$</td>
</tr>
<tr>
<td>$\Sigma; P; \Delta \rightarrow \forall x : \iota. , G$</td>
<td>$\forall R$</td>
</tr>
<tr>
<td>$\Sigma; P; \Delta \rightarrow \exists x : \iota. , G$</td>
<td>$\exists R$</td>
</tr>
<tr>
<td>$\Sigma; P; \Delta \rightarrow \exists x : \iota. , G$</td>
<td>$\forall R$</td>
</tr>
<tr>
<td>$\Sigma; P; \Delta \rightarrow \exists x : \iota. , G$</td>
<td>$\forall R$</td>
</tr>
<tr>
<td>$\Sigma; P; \Delta \rightarrow \exists x : \iota. , G$</td>
<td>$\forall R$</td>
</tr>
</tbody>
</table>

**Figure:** Uniform Proof, with the field $\Delta$ for a coinductive invariant, and the relation $\equiv$ for equality between guarded terms.
Guarding Mechanism

\[ \Sigma; P; \Delta \overset{D}{\Rightarrow} A \quad P \ni D \notin \Delta \quad \text{DECIDE} \langle \rangle \]

\[ \Sigma; P; \Delta \Rightarrow \langle A \rangle \]

\[ c : \iota, \Sigma; P; \Delta \Rightarrow \langle M[c/x] \rangle \quad c : \iota \notin \Sigma \quad \forall R \langle \rangle \]

\[ \Sigma; P; \Delta \Rightarrow \langle \forall x : \iota. M \rangle \]

\[ \Sigma; P; \Delta \Rightarrow \langle M_1 \rangle \quad \Sigma; P; \Delta \Rightarrow \langle M_2 \rangle \quad \land R \langle \rangle \quad \Sigma; P; M_1; \Delta \Rightarrow \langle M_2 \rangle \quad \rightarrow R \langle \rangle \]

\[ \Sigma; P; \Delta \Rightarrow \langle M_1 \land M_2 \rangle \]

\[ \Sigma; P; \Delta \Rightarrow \langle M_1 \rightarrow M_2 \rangle \]

**Figure:** Guarding Mechanism
Soundness Properties: w.r.t Herbrand Model

CUP is sound w.r.t the greatest fixed-point model $M_\nu$.

**Theorem**

If $\Sigma; P \leftrightarrow \varphi$ then $M_\nu \models \varphi$.

**Proof Sketch.**

- A coinductive uniform proof is a template.
- Using certain substitutions involved in the proof,
- an infinite amount of substitutions can be generated,
- which can instantiate the template into an infinite amount of instances
- The infinite SLD-derivation can be obtained by assembling these instances.
Soundness Properties: w.r.t Herbrand Model

CUP is sound w.r.t the greatest fixed-point model $M_\nu$.

**Theorem**

*If $\Sigma; P \vDash \varphi$ and $\Sigma; P, \varphi \vDash \psi$, then $M_\nu \models \psi$ — provided $\varphi$ either has no $\forall$ or has no $\rightarrow$.*

**Proof Sketch.**

Since $\Sigma; P \vDash \varphi$, we have $M_\nu \models \varphi$. Let $M_\nu'$ be the greatest fixed-point model of $P \cup \{\varphi\}$. Since $\Sigma; P, \varphi \vDash \psi$, we have $M_\nu' \models \psi$. We show that $M_\nu = M_\nu'$.

- If $\varphi$ involves both $\rightarrow$ and $\forall$, we may still use $\varphi$ as a lemma, provided some further conditions are satisfied.
Soundness Properties: w.r.t iFOL

CUP is sound w.r.t intuitionistic sequent calculus extended with later modality (iFOL)

Definition
The formulae of the logic iFOL over Σ are well formed formulae extended with the following rule. Conversion (≡) extends to these formulae in the obvious way.

\[
\Sigma; \Gamma \vdash \varphi
\]

\[
\Sigma; \Gamma \vdash \square \varphi
\]

Definition
Γ | Δ ⊬ ϕ means the formula ϕ is provable in context Γ w.r.t the set Δ of formulae.
Soundness Properties: w.r.t iFOL

\[ \Sigma; \Gamma \vdash \Delta \quad \varphi \in \Delta \]
\[ \Gamma \mid \Delta \vdash \varphi \]

(Proj)

\[ \Gamma \mid \Delta \vdash \varphi \quad \Gamma \mid \Delta \vdash \psi \]
\[ \Gamma \mid \Delta \vdash \varphi \land \psi \]

(^-I)

\[ \Gamma \mid \Delta \vdash \varphi_i \quad i \in \{1, 2\} \]
\[ \Gamma \mid \Delta \vdash \varphi_1 \lor \varphi_2 \]

(^-I)

\[ \Gamma \mid \Delta \vdash \varphi \quad \Gamma \mid \Delta \vdash \psi \]
\[ \Gamma \mid \Delta \vdash \varphi \rightarrow \psi \]

(\rightarrow-I)

\[ \Gamma, x : \tau \mid \Delta \vdash \varphi \]
\[ x : \tau \notin \Gamma \]
\[ \Gamma \mid \Delta \vdash \forall x : \tau. \varphi \]

(^-I)

\[ \Sigma; \Gamma \vdash_{(m; n)} M : \tau \]
\[ \Gamma \mid \Delta \vdash [M/x] \]
\[ \Gamma \mid \Delta \vdash \forall x : \tau. \varphi \]

(^-E)

\[ \Sigma; \Gamma \vdash_{(m; n)} M : \tau \]
\[ \Gamma \mid \Delta \vdash \varphi [M/x] \]
\[ \Gamma \mid \Delta \vdash \exists x : \tau. \varphi \]

(^-E)

\[ \Gamma, x : \tau \mid \Delta, \varphi \vdash \psi \]
\[ x : \tau \notin \Gamma \]
\[ \Gamma \mid \Delta \vdash \exists x : \tau. \varphi \vdash \psi \]

(\exists-E)

Intuitionistic Rules for Standard Connectives

\[ \Gamma \mid \Delta \vdash \varphi \]
\[ \Gamma \mid \Delta \vdash \varphi \rightarrow \psi \]

(Next)

(Next)

\[ \Gamma \mid \Delta \vdash \varphi \rightarrow \psi \]
\[ \Gamma \mid \Delta \vdash \varphi \rightarrow \psi \]

(Mon)

\[ \Gamma \mid \Delta \vdash \varphi \rightarrow \psi \]

(Löb)

Rules for the Later Modality
Soundness Properties: w.r.t iFOL

**Definition**
Given a Horn clause \( \varphi \) of the shape \( \forall \vec{x}. (A_1 \land \cdots \land A_n) \rightarrow A \), we define its **guarding** \( \overline{\varphi} \) to be \( \forall \vec{x}. (\Box A_1 \land \cdots \land \Box A_n) \rightarrow A \). For a collection \( P \) of Horn clauses, we define its guarding \( \overline{P} \) by guarding each formula in \( P \).

**Theorem**
If \( \Sigma; P \vdash \varphi \) then \( \emptyset \vdash \overline{P} \vdash \varphi \).

**Proof Sketch.**
We do case analysis with an inductive argument.
Summary

Introduction
Background
  Herbrand Models
  CoLP
  Precor
  Limitations
Coinductive Uniform Proof
  Motivation
  Overview
  Terms and Formulae
    Overview of Term Syntax
    The Type System

Signature and Context
  Well Formed Terms
  Guarded Terms
  Well Formed Formulae
  Hereditary Harrop Formula
  Equivalence Relation

CUP Rules
  Coinductive Proof Principle
  Uniform Proof
  Guarding Mechanism

Soundness Properties
  w.r.t Herbrand Model
  w.r.t iFOL
Acknowledgment

We thank Dr Murdoch James Gabbay for his suggestions to improve these slides!