Uniform Inductive Reasoning in Transitive Closure Logic via Infinite Descent

Liron Cohen \(^1\)  Reuben N. S. Rowe \(^2\)
Computer Science Logic
Wednesday 5\(^{th}\) September 2018, Birmingham, UK

\(^1\)Dept of Computer Science, Cornell University, Ithaca, NY, USA

\(^2\)School of Computing, University of Kent, Canterbury, UK
We trace syntactic elements through judgements.

At certain points, there is a notion of ‘progression’.

Each infinite path must admit some infinite descent.

This global trace condition is an $\omega$-regular property, i.e., decidable using Büchi automata.
We trace syntactic elements through judgements.

At certain points, there is a notion of 'progression'.

Each infinite path must admit some infinite descent.

This global trace condition is an $\omega$-regular property.

i.e. decidable using Büchi automata.
We trace syntactic elements $\tau$ through judgements.
We trace syntactic elements $\tau$ through judgements.

At certain points, there is a notion of ‘progression’.
We trace syntactic elements $\tau$ through judgements.

- At certain points, there is a notion of ‘progression’
- Each infinite path must admit some infinite descent
• We trace syntactic elements $\tau$ through judgements
  • At certain points, there is a notion of ‘progression’
• Each infinite path must admit some infinite descent
• This global trace condition is an $\omega$-regular property
  • i.e. decidable using Büchi automata
Assume for contradiction that the conclusion is invalid.

Local soundness (counter-models $M_1; M_2; M_3; \ldots$)

We demonstrate a mapping into well-founded $(D; <)$ s.t.

$J_1 \Rightarrow \ldots \Rightarrow J_2 \Rightarrow J_3 \Rightarrow \ldots$

for progression points

Global trace condition: infinitely descending chain in $D$
Non-well-founded Proofs: Soundness via Infinite Descent

- Assume for contradiction that the conclusion is invalid
• Assume for contradiction that the conclusion is invalid
  • Local soundness $\Rightarrow$ counter-models $M_1, M_2, M_3, \ldots$
Assume for contradiction that the conclusion is invalid

- Local soundness $\Rightarrow$ counter-models $M_1, M_2, M_3, \ldots$
- We demonstrate a mapping into well-founded $(D, <)$ s.t.
  - $\llbracket M_1 \rrbracket_{J_1[\tau_1]} \leq \llbracket M_2 \rrbracket_{J_2[\tau_2]} \leq \llbracket M_3 \rrbracket_{J_3[\tau_3]} \leq \ldots$
Non-well-founded Proofs: Soundness via Infinite Descent

- Assume for contradiction that the conclusion is invalid
  - Local soundness \( \Rightarrow \) counter-models \( M_1, M_2, M_3, \ldots \)
  - We demonstrate a mapping into well-founded \((D, <)\) s.t.
    - \( \llbracket M_1 \rrbracket_{J_1[\tau_1]} \leq \llbracket M_2 \rrbracket_{J_2[\tau_2]} \leq \llbracket M_3 \rrbracket_{J_3[\tau_3]} \leq \ldots \)
    - \( \llbracket M_2 \rrbracket_{J_2[\tau_2]} < \llbracket M_3 \rrbracket_{J_3[\tau_3]} \) for progression points
Non-well-founded Proofs: Soundness via Infinite Descent

- Assume for contradiction that the conclusion is invalid
  - Local soundness $\Rightarrow$ counter-models $M_1, M_2, M_3, \ldots$
  - We demonstrate a mapping into well-founded $(D, <)$ s.t.
    - $[M_1]_{J_1[\tau_1]} \leq [M_2]_{J_2[\tau_2]} \leq [M_3]_{J_3[\tau_3]} \leq \cdots$
    - $[M_2]_{J_2[\tau_2]} < [M_3]_{J_3[\tau_3]}$ for progression points
  - Global trace condition $\Rightarrow$ infinitely descending chain in $D$!
Why Study Non-well-founded Proof Theory?

Non-well-founded/cyclic proof theory allows to:

- Obtain (cut-free) completeness results
  \(\mu\)-calculus: Fortier\&Santocanale, Afshari\&Leigh, Doumane Et Al.
  Kleene Algebra: Das\&Pous

- Effectively search for proofs of inductive properties

- Automatically verify properties of programs
  [Brotherston, Bornat, Calcagno, Gorogiannis, Peterson, R, Tellez]

- Formally study explicit induction vs infinite descent
  \(\mu\)-calculus: Santocanale, Sprenger\&Dam, Baelde Et Al., Nollet Et Al.
  Ind. Defs: Brotherston\&Simpson, Berardi\&Tatsuta
  Arithmetic: Simpson, Das
Example: Martin-Löf-style Inductive Predicates in FOL

- We give productions for each ‘inductive’ predicate $P_i$

\[
Q_1(s_1) \ldots Q_n(s_n) \\
\vdash P_i(t)
\]

- We take the smallest interpretation closed under the rules

\[
\begin{array}{cccc}
\hline
N & N \times & O & E \\
0 & N s x & O x & E x \\
\hline
N & N s x & E & O s x \\
0 & \hline
\end{array}
\]

\[
\begin{align*}
\llbracket N \rrbracket &= \{0, s0, ss0, \ldots, s^n 0, \ldots \} \\
\llbracket E \rrbracket &= \{0, ss0, \ldots, s^{2n} 0, \ldots \} \\
\llbracket O \rrbracket &= \{s0, \ldots, s^{2n+1} 0, \ldots \}
\end{align*}
\]
Example: Martin-Löf-style Inductive Predicates in FOL

- We give productions for each ‘inductive’ predicate $P_i$

\[
\begin{align*}
Q_1(s_1) & \quad \ldots \quad Q_n(s_n) \\
\hline
P_i(t)
\end{align*}
\]

- We take the smallest interpretation closed under the rules

\[
\begin{array}{cccc}
\text{N} & \text{N} & \text{O} & \text{E} \\
0 & \times & 0 & \times \\
\hline
\text{N} & \text{N} & \text{E} & \text{E} \\
0 & \times & 0 & \times \\
\hline
\text{N} & \text{O} & \text{E} & \text{O} \\
0 & \times & 0 & \times \\
\hline
\text{N} & \text{E} & \text{E} & \text{O} \\
s_0 & \times & s_0 & \times \\
\hline
\text{N} & \text{O} & \text{E} & \text{O} \\
s_2 & \times & s_2 & \times \\
\hline
\text{N} & \text{E} & \text{E} & \text{O} \\
s_2 & \times & s_2 & \times \\
\hline
\text{N} & \text{O} & \text{E} & \text{O} \\
s_2 & \times & s_2 & \times \\
\hline
\text{N} & \text{E} & \text{E} & \text{O} \\
s_2 & \times & s_2 & \times \\
\hline
\end{array}
\]

\[
\begin{align*}
\llbracket N \rrbracket_0 &= \{ \} \\
\llbracket E \rrbracket_0 &= \{ \} \\
\llbracket O \rrbracket_0 &= \{ \}
\end{align*}
\]
Example: Martin-Löf-style Inductive Predicates in FOL

• We give productions for each ‘inductive’ predicate $P_i$

\[
\frac{Q_1(s_1) \quad \ldots \quad Q_n(s_n)}{P_i(t)}
\]

• We take the smallest interpretation closed under the rules

<table>
<thead>
<tr>
<th>$N$</th>
<th>$Nx$</th>
<th>$E$</th>
<th>$Ox$</th>
<th>$Ex$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$N0$</td>
<td>$Nsx$</td>
<td>$E0$</td>
<td>$Esx$</td>
<td>$O$sx</td>
</tr>
</tbody>
</table>

$\{N\}_1 = \{0, \}$

$\{E\}_1 = \{0, \}$

$\{O\}_1 = \{}$
Example: Martin-Löf-style Inductive Predicates in FOL

- We give productions for each ‘inductive’ predicate $P_i$

$$Q_1(s_1) \quad \ldots \quad Q_n(s_n)$$

\[
P_i(t)
\]

- We take the smallest interpretation closed under the rules

<table>
<thead>
<tr>
<th>N 0</th>
<th>N x</th>
<th>E 0</th>
<th>O x</th>
<th>E x</th>
</tr>
</thead>
<tbody>
<tr>
<td>N s x</td>
<td>E s x</td>
<td>O s x</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\[
[N]_2 = \{ 0, s0, \}
\]

\[
[E]_2 = \{ 0, \}
\]

\[
[O]_2 = \{ s0, \}
\]
Example: Martin-Löf-style Inductive Predicates in FOL

• We give productions for each ‘inductive’ predicate $P_i$

$$\frac{Q_1(s_1)}{P_i(t)} \quad \ldots \quad \frac{Q_n(s_n)}{P_i(t)}$$

• We take the smallest interpretation closed under the rules

\[
\begin{align*}
\text{N} & \quad \text{O} \\
\text{N} & \quad \text{N} \times \text{N} \times \\
\text{E} & \quad \text{O} \\
\text{E} & \quad \text{E} \times \text{E} \times \\
\text{O} & \quad \text{O} \times \text{O} \times \\
\end{align*}
\]

$$\begin{align*}
\llbracket \text{N} \rrbracket_3 &= \{0, s0, ss0, \ldots\} \\
\llbracket \text{E} \rrbracket_3 &= \{0, ss0, \ldots\} \\
\llbracket \text{O} \rrbracket_3 &= \{s0, \ldots\}
\end{align*}$$
Example: Martin-Löf-style Inductive Predicates in FOL

- We give productions for each ‘inductive’ predicate $P_i$

$$
\begin{align*}
Q_1(\vec{s}_1) & \quad \ldots \quad Q_n(\vec{s}_n) \\
\hline
P_i(\vec{t})
\end{align*}
$$

- We take the smallest interpretation closed under the rules

<table>
<thead>
<tr>
<th>$N \ o$</th>
<th>$N \ x$</th>
<th>$N \ s \ x$</th>
<th>$E \ o$</th>
<th>$E \ s \ x$</th>
<th>$O \ s \ x$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$N \ o$</td>
<td>$N \ x$</td>
<td>$E \ o$</td>
<td>$E \ s \ x$</td>
<td>$O \ s \ x$</td>
<td></td>
</tr>
</tbody>
</table>

\[
\begin{align*}
\llbracket N \rrbracket_\omega &= \{ 0, s0, ss0, \ldots, s^n 0, \ldots \} \\
\llbracket E \rrbracket_\omega &= \{ 0, ss0, \ldots, s^{2n} 0, \ldots \} \\
\llbracket O \rrbracket_\omega &= \{ s0, \ldots, s^{2n+1} 0, \ldots \}
\end{align*}
\]
Example: A Cyclic Proof

\[ \Rightarrow N \, 0 \]
\[ N \, x \Rightarrow N \, sx \]
\[ \Rightarrow E \, 0 \]
\[ O \, x \Rightarrow E \, sx \]
\[ E \, x \Rightarrow O \, sx \]

\[ \frac{}{ \vdash N \, 0} \]
\[ \frac{}{ \vdash N \, x} \]
\[ \frac{\frac{}{ \vdash N \, 0}}{\vdash N \, x} \]
\[ \frac{\frac{}{ \vdash N \, 0}}{\vdash N \, x} \]

\[ y = sz, Ez \vdash Ny \]
\[ \frac{}{\vdash Ny} \]
\[ O \, y \vdash N \, y \]
\[ \frac{}{\vdash N \, sy} \]
\[ \frac{}{\vdash N \, sy} \]
\[ \frac{}{\vdash N \, x} \]
\[ \frac{}{\vdash N \, x} \]

\[ \frac{}{\vdash N \, 0} \]
\[ \frac{}{\vdash N \, 0} \]
\[ \frac{}{\vdash N \, 0} \]
\[ \frac{}{\vdash N \, 0} \]

\[ Ex \vdash N \, x \]
\[ Ez \vdash N \, z \]
\[ Ez \vdash N \, sz \]
\[ \frac{}{\vdash N \, z} \]
\[ \frac{}{\vdash N \, sy} \]
\[ \frac{}{\vdash N \, x} \]
\[ \frac{}{\vdash N \, x} \]
\[ \frac{}{\vdash N \, x} \]

\[ (N \, R_1) \]
\[ (N \, R_2) \]
\[ (N \, R_2) \]
\[ (N \, R_2) \]

\[ (\Rightarrow) \]
\[ (\Rightarrow) \]
\[ (\Rightarrow) \]
\[ (\Rightarrow) \]

\[ (=L) \]
\[ (=L) \]
\[ (=L) \]
\[ (=L) \]

\[ (Subst) \]
\[ (Case \, O) \]
\[ (Case \, E) \]
Example: A Cyclic Proof

\[ \begin{align*}
\Rightarrow & \quad N \ 0 \\
N \ x & \Rightarrow \quad N \ sx \\
\Rightarrow & \quad E \ 0 \\
O \ x & \Rightarrow \quad E \ sx \\
E \ x & \Rightarrow \quad O \ sx
\end{align*} \]

\[ \begin{align*}
Ex & \vdash \quad N \ x \\
(Ez \vdash \quad N \ sz) & \quad (N \ R_2) \\
Ez & \vdash \quad N \ sz \\
\Rightarrow & \quad (=L) \\
y = sz, Ez \vdash \quad N \ y & \quad (Case \ O) \\
O \ y & \vdash \quad N \ y \\
\Rightarrow & \quad (=L) \\
x = 0 \vdash \quad N \ x & \quad (Case \ E) \\
x = sy, O \ y & \vdash \quad N \ x
\end{align*} \]
Example: A Cyclic Proof

\[ \Rightarrow N \, 0 \]
\[ N \, x \Rightarrow N \, sx \]
\[ \Rightarrow E \, 0 \]
\[ O \, x \Rightarrow E \, sx \]
\[ E \, x \Rightarrow O \, sx \]

Left unfolding rule

\[
\begin{align*}
&\text{Ex} \vdash N \, x \\
&\text{(Subst)} \\
&Ez \vdash N \, z \\
&\text{(N R}_2) \\
&Ez \vdash N \, sz \\
&\text{(=L)} \\
&y = sz, Ez \vdash N \, y \\
&\text{(Case O)} \\
&Oy \vdash N \, y \\
&\text{(N R}_2) \\
&Oy \vdash N \, sy \\
&\text{(=L)} \\
&x = sy, Oy \vdash N \, x \\
&\text{(Case E)}
\end{align*}
\]

\[ Ex \vdash N \, x \]
Example: A Cyclic Proof

Right unfolding rule

\[ \Rightarrow N \, 0 \]
\[ N \, x \Rightarrow N \, sx \]
\[ \Rightarrow E \, 0 \]
\[ O \, x \Rightarrow E \, sx \]
\[ E \, x \Rightarrow O \, sx \]

\[ \vdash N \, 0 \]
\[ (N \, R_1) \]
\[ x = 0 \vdash N \, x \]
\[ (=L) \]

\[ Ex \vdash N \, x \]
\[ Ez \vdash N \, z \]
\[ Ez \vdash N \, sz \]
\[ (=L) \]

\[ y = sz, Ez \vdash N \, y \]
\[ (Case \, O) \]
\[ O \, y \vdash N \, y \]
\[ (N \, R_2) \]
\[ O \, y \vdash N \, sy \]
\[ (=L) \]

\[ x = sy, O \, y \vdash N \, x \]
\[ (Case \, E) \]

\[ Ex \vdash N \, x \]
Example: A Cyclic Proof

\[ \Rightarrow N \, 0 \]
\[ N \, x \Rightarrow N \, sx \]
\[ \Rightarrow E \, 0 \]
\[ O \, x \Rightarrow E \, sx \]
\[ E \, x \Rightarrow O \, sx \]

\[ \begin{align*}
Ex \vdash N \, x \\
Ez \vdash N \, z \\
Ez \vdash N \, sz \\
y = sz, Ez \vdash N \, y \\
Oy \vdash N \, y \\
Oy \vdash N \, sy \\
x = 0 \vdash N \, x \\
x = sy, Oy \vdash N \, x \\
Ex \vdash N \, x
\end{align*} \]
Example: A Cyclic Proof

\[ N \Rightarrow N 0 \]
\[ N x \Rightarrow N sx \]
\[ \Rightarrow E 0 \]
\[ O x \Rightarrow E sx \]
\[ E x \Rightarrow O sx \]

\[ Ex \vdash N x \quad \text{(Subst)} \]
\[ Ez \vdash N z \quad \text{\( N R_2 \)} \]
\[ Ez \vdash N sz \quad \text{\( = L \)} \]
\[ y = sz, Ez \vdash N y \quad \text{(Case O)} \]
\[ O y \vdash N y \quad \text{\( N R_2 \)} \]
\[ O y \vdash N sy \quad \text{\( = L \)} \]
\[ x = 0 \vdash N x \quad \text{\( = L \)} \]
\[ x = sy, O y \vdash N x \quad \text{\( = L \)} \]
\[ Ex \vdash N x \quad \text{(Case E)} \]
Example: A Cyclic Proof

\[ \begin{align*}
\Rightarrow & \quad N \, 0 \\
N \, x & \Rightarrow \quad N \, sx \\
\Rightarrow & \quad E \, 0 \\
O \, x & \Rightarrow \quad E \, sx \\
E \, x & \Rightarrow \quad O \, sx
\end{align*} \]

\[ \begin{align*}
Ex \vdash N \, x & \quad \text{(Subst)} \\
Ez \vdash N \, z & \quad \text{(N R)} \\
Ez \vdash N \, sz & \\
\quad \vdash N \, 0 & \quad (=L) \\
y = sz, Ez \vdash N \, y & \quad \text{(Case O)} \\
Oy \vdash N \, y & \quad \text{(N R)} \\
Oy \vdash N \, sy & \\
x = 0 \vdash N \, x & \quad (=L) \\
x = sy, Oy \vdash N \, x & \quad (=L) \\
Ex \vdash N \, x & \quad \text{(Case E)}
\end{align*} \]
Example: A Cyclic Proof

\[
\begin{align*}
  &\Rightarrow N\ 0 \\
  &N\ x \Rightarrow N\ sx \\
  &\Rightarrow E\ 0 \\
  &O\ x \Rightarrow E\ sx \\
  &E\ x \Rightarrow O\ sx
\end{align*}
\]

\[
\begin{align*}
  Ex &\vdash Nx \\
  Ez &\vdash Nz \\
  Ez &\vdash Nsz \\
  E &\vdash sz \Rightarrow Nz \\
  E &\vdash Nsy \\
  y &\Rightarrow sx, Ez \vdash Ny \\
  Oy &\vdash Ny \\
  Oy &\vdash Nsy \\
  Ox &\vdash Nsx \\
  Ex &\vdash Nx \\
  \end{align*}
\]
Cyclic Proof vs Explicit Induction

- To reason explicitly by induction is more complex, involving an **induction formula** $F$

\[ \Gamma \vdash \text{IND}_{Q_i}(F) \quad (\forall Q_i \text{ mutually recursive with } P) \quad \Gamma, F(\tilde{t}) \vdash \Delta \]

\[ \Gamma, F(\tilde{t}) \vdash \Delta \]

- E.g. the productions $\Rightarrow N 0$ and $N x \Rightarrow N sx$ give

\[ \Gamma \vdash F(0) \quad \Gamma, F(x) \vdash F(sx) \quad \Gamma, F(t) \vdash \Delta \]

\[ \Gamma, N t \vdash \Delta \]

- Implicit induction using **unfolding** conceptually simpler
  - Induction schemes captured using cycles
Non-well-founded Proofs: Some Meta-theory

For FOL with Inductive Definitions:

- Non-well-founded proof system LKID$^\omega$ sound and cut-free complete for standard semantics
- Explicit induction system LKID sound and cut-free complete for a Henkin-style semantics
- Cyclic system CLKID$^\omega$ subsumes explicit induction
  [Brotherston & Simpson, LICS’07, JL&C’11]
- CLKID$^\omega$ and LKID equivalent under arithmetic
  [Berardi & Tatsuta, LICS’17]
  [Simpson, FoSSaCS’17]
- CLKID$^\omega$ and LKID not equivalent in general (2-Hydra counterexample)
  [Berardi & Tatsuta, FoSSaCS’17]
Transitive Closure Logic

Transitive Closure (TC) Logic extends FOL with formulas:

\[
(\text{RTC}_{x,y} \varphi)(s, t)
\]

- \( \varphi \) is a formula
- \( x \) and \( y \) are distinct variables (which become bound in \( \varphi \))
- \( s \) and \( t \) are terms

whose intended meaning is an infinite disjunction

\[
s = t \lor \varphi[s/x, t/y] \\
\lor (\exists w_1 . \varphi[s/x, w_1/y] \land \varphi[w_1/x, t/y]) \\
\lor (\exists w_1, w_2 . \varphi[s/x, w_1/y] \land \varphi[w_1/x, w_2/y] \land \varphi[w_2/x, t/y]) \\
\lor \ldots
\]
Transitive Closure Logic

The formal semantics:

- $M$ is a (standard) first-order model with domain $D$
- $v$ is a valuation of terms in $M$:

$$M, v \models (RTC_{x,y} \varphi)(s, t)$$
Transitive Closure Logic

The formal semantics:

- $M$ is a (standard) first-order model with domain $D$
- $v$ is a valuation of terms in $M$:

$$M, v \models (RTC_{x, y} \varphi)(s, t) \iff \exists a_0, \ldots, a_n \in D$$
Transitive Closure Logic

The formal semantics:

- $M$ is a (standard) first-order model with domain $D$
- $v$ is a valuation of terms in $M$:

$$M, v \models (RTC_{x,y} \varphi)(s, t) \iff \exists a_0, \ldots, a_n \in D . v(s) = a_0 \land v(t) = a_n$$
Transitive Closure Logic

The formal semantics:

- $M$ is a (standard) first-order model with domain $D$
- $v$ is a valuation of terms in $M$:

$$M, v \models (RTC_{x,y} \varphi)(s, t) \iff$$
$$\exists a_0, \ldots, a_n \in D . v(s) = a_0 \land v(t) = a_n$$
$$\land M, v[x := a_i, y := a_{i+1}] \models \varphi \quad \text{for all } i < n$$
Example: Arithmetic in TC

- Take a signature $\Sigma = \{0, s\} + \text{equality}$

$$\text{Nat}(x) \equiv (RTC_{v,w} s v = w)(0, x)$$
Example: Arithmetic in TC

- Take a signature $\Sigma = \{0, s\} + \text{equality}$

$$\text{Nat}(x) \equiv (RTC_{v,w} sv = w)(0, x)$$

$$x \leq y \equiv (RTC_{v,w} sv = w)(x, y)$$
Example: Arithmetic in TC

- Take a signature $\Sigma = \{0, s\} + \text{equality and pairing}$

\[
\text{Nat}(x) \equiv (RTC_{v,w} s v = w)(0, x)
\]

\[
x \leq y \equiv (RTC_{v,w} s v = w)(x, y)
\]

“$x = y + z$” $\equiv$

\[
(RTC_{v,w} \exists n_1, n_2 . \ v = \langle n_1, n_2 \rangle \land w = \langle sn_1, sn_2 \rangle)(\langle 0, y \rangle, \langle z, x \rangle)
\]
Example: Arithmetic in TC

• Take a signature $\Sigma = \{0, s\} + \text{equality and pairing}$

$$\text{Nat}(x) \equiv (RTC_{v,w} sv = w)(0, x)$$

$$x \leq y \equiv (RTC_{v,w} sv = w)(x, y)$$

“$x = y + z$” $\equiv$

$$(RTC_{v,w} \exists n_1, n_2 \cdot v = \langle n_1, n_2 \rangle \land w = \langle sn_1, sn_2 \rangle)(\langle 0, y \rangle, \langle z, x \rangle)$$
Example: Arithmetic in TC

• Take a signature $\Sigma = \{0, s\}$ + equality and pairing

$\text{Nat}(x) \equiv (RTC_{v,w} sv = w)(0, x)$

$x \leq y \equiv (RTC_{v,w} sv = w)(x, y)$

“$x = y + z$” $\equiv$

$(RTC_{v,w} \exists n_1, n_2 . v = \langle n_1, n_2 \rangle \land w = \langle sn_1, sn_2 \rangle)((0, y), (z, x))$
Example: Arithmetic in TC

- Take a signature $\Sigma = \{0, s\} + \text{equality and pairing}$

\[
\text{Nat}(x) \equiv (RTC_{v,w} \ s v = w)(0, x)
\]

\[
x \leq y \equiv (RTC_{v,w} \ s v = w)(x, y)
\]

“$x = y + z$” $\equiv$

\[
(RTC_{v,w} \ \exists n_1, n_2 . \ v = \langle n_1, n_2 \rangle \land w = \langle sn_1, sn_2 \rangle)(\langle 0, y \rangle, \langle z, x \rangle)
\]
Example: Arithmetic in TC

- Take a signature $\Sigma = \{0, s\} \cup \text{equality and pairing}$

\[
\text{Nat}(x) \equiv (RTC_{v,w} \; sv = w)(0, x)
\]

\[
x \leq y \equiv (RTC_{v,w} \; sv = w)(x, y)
\]

"$x = y + z$" $\equiv$

\[
(RTC_{v,w} \; \exists n_1, n_2 \cdot v = \langle n_1, n_2 \rangle \land w = \langle sn_1, sn_2 \rangle)(\langle 0, y \rangle, \langle z, x \rangle)
\]
Example: Arithmetic in TC

- Take a signature $\Sigma = \{0, s\}$ + equality and pairing

$$\text{Nat}(x) \equiv (RTC_{v,w} sv = w)(0, x)$$

$$x \leq y \equiv (RTC_{v,w} sv = w)(x, y)$$

"$x = y + z$" $\equiv$

$$(RTC_{v,w} \exists n_1, n_2 . v = \langle n_1, n_2 \rangle \land w = \langle sn_1, sn_2 \rangle)(\langle 0, y \rangle, \langle z, x \rangle)$$

- The following characterise natural numbers in TC:

$$\forall x . sx \neq 0$$

$$\forall x, y . s(x) = s(y) \to x = y$$

$$\forall x . \text{Nat}(x)$$
Why Study TC and its Non-well-founded Proof Theory?

- Provides a uniform way to express inductive definitions
  - Single framework for modelling many areas of CS
  - Better for automated reasoning?
- It is a *minimal*, yet *expressive*, extension of FOL

**Theorem (Avron ’03, Thm. 3)**

All finitely inductively definable relations\(^\dagger\) are definable in TC.

A. Avron, *Transitive Closure and the Mechanization of Mathematics*.

- Alternative setting for studying cyclic vs explicit induction
  - No need to ‘choose’ predicates up-front
  - Uniformity makes meta-theory more straightforward
  - Displays some subtle but important differences with FOL+ID

\(^\dagger\)as formalised in: S. Feferman, *Finitary Inductively Presented Logics*, 1989
Implicit and Explicit Induction Rules for TC

**reflexivity**

\[ \Gamma \vdash (RTC_{x,y} \varphi)(t, t) \]

**step**

\[ \begin{align*} 
\Gamma & \vdash \Delta, (RTC_{x,y} \varphi)(s, r) & \Gamma & \vdash \Delta, \varphi[r/x, t/y] \\
\hline
\Gamma & \vdash \Delta, (RTC_{x,y} \varphi)(s, t) 
\end{align*} \]

**case-split**

\[ \begin{align*} 
\Gamma, s = t & \vdash \Delta & \Gamma, (RTC_{x,y} \varphi)(s, z), \varphi[z/x, t/y] & \vdash \Delta \\
\hline
\Gamma, (RTC_{x,y} \varphi)(s, t) & \vdash \Delta 
\end{align*} \]
Implicit and Explicit Induction Rules for TC

**reflexivity**

\[ \Gamma \vdash (RTC_{x,y} \varphi)(t, t) \]

**step**

\[ \Gamma \vdash \Delta, (RTC_{x,y} \varphi)(s, r) \quad \Gamma \vdash \Delta, \varphi[r/x, t/y] \]

\[ \Gamma \vdash \Delta, (RTC_{x,y} \varphi)(s, t) \]

**case-split**

\[ \Gamma, s = t \vdash \Delta \quad \Gamma, (RTC_{x,y} \varphi)(s, z), \varphi[z/x, t/y] \vdash \Delta \]

\[ \Gamma, (RTC_{x,y} \varphi)(s, t) \vdash \Delta \] (z fresh)
Implicit and Explicit Induction Rules for TC

reflexivity

\[ \Gamma \vdash (RTC_{x,y} \varphi)(t, t) \]

step

\[ \Gamma \vdash \Delta, (RTC_{x,y} \varphi)(s, r) \quad \Gamma \vdash \Delta, \varphi[r/x, t/y] \]

\[ \Gamma \vdash \Delta, (RTC_{x,y} \varphi)(s, t) \]

case-split

\[ \Gamma \vdash \Delta, (RTC_{x,y} \varphi)(s, t) \vdash \Delta \]

induction

\[ x \not\in \text{fv}(\Gamma, \Delta) \text{ and } y \not\in \text{fv}(\Gamma, \Delta, \psi) \]
Proof-theoretic Results for TC

• Non-well-founded system $\text{RTC}_G^\omega$ sound + cut-free complete for standard semantics

• Explicit induction system $\text{RTC}_G$ sound + cut-free complete for a Henkin-style semantics

• Cyclic system subsumes explicit induction
  
  $\text{RTC}_G \subseteq \text{NCRTC}_G^\omega$ (non-overlapping cycles) $\subseteq \text{CRTC}_G^\omega$
Proof-theoretic Results for TC

- Non-well-founded system $\text{RTC}_G^{\omega}$ sound + cut-free complete for standard semantics
- Explicit induction system $\text{RTC}_G$ sound + cut-free complete for a Henkin-style semantics
- Cyclic system subsumes explicit induction
  $\text{RTC}_G \subseteq \text{NCRTC}_G^{\omega}$ (non-overlapping cycles) $\subseteq \text{CRTC}_G^{\omega}$
- Systems with arithmetic are equivalent
Proof-theoretic Results for TC

- Non-well-founded system $\text{RTC}_G^\omega$ sound + cut-free complete for standard semantics
- Explicit induction system $\text{RTC}_G$ sound + cut-free complete for a Henkin-style semantics
- Cyclic system subsumes explicit induction $\text{RTC}_G \subseteq \text{NCRTC}_G^\omega$ (non-overlapping cycles) $\subseteq \text{CRTC}_G^\omega$
- Systems with arithmetic are equivalent

$\text{RTC}_G + A \leftrightarrow \text{PA}_G \leftrightarrow \text{CRTC}_G^\omega + A$

C & Avron, ’15
Proof-theoretic Results for TC

- Non-well-founded system $RTC_G^\omega$ sound + cut-free complete for standard semantics
- Explicit induction system $RTC_G$ sound + cut-free complete for a Henkin-style semantics
- Cyclic system subsumes explicit induction
  \[ RTC_G \subseteq NCRTC_G^\omega \text{ (non-overlapping cycles)} \subseteq CRTC_G^\omega \]
- Systems with arithmetic are equivalent
Proof-theoretic Results for TC

- Non-well-founded system RTC₆₆ sound + cut-free complete for standard semantics
- Explicit induction system RTC₆ sound + cut-free complete for a Henkin-style semantics
- Cyclic system subsumes explicit induction
  \[ RTC₆ \subseteq NCRTC₆ (\text{non-overlapping cycles}) \subseteq CRTC₆ \]
- Systems with arithmetic are equivalent

\[ RTC₆ + A \rightarrow PA₆ \rightarrow CA₆ \rightarrow CRTC₆ + A \]

Simpson, ’17
C&R
C & Avron, ’15
Proof-theoretic Results for TC

- Non-well-founded system $RTC_G^\omega$ sound + cut-free complete for standard semantics
- Explicit induction system $RTC_G$ sound + cut-free complete for a Henkin-style semantics
- Cyclic system subsumes explicit induction
  \[ RTC_G \subseteq NCRTC_G^\omega \text{ (non-overlapping cycles)} \subseteq CRTC_G^\omega \]
- Systems with arithmetic are equivalent
- 2-Hydra counterexample does not show $RTC_G \subsetneq CRTC_G^\omega$
  - Relies on not being able to express ordering on numbers
  - TC allows all inductive definitions ‘at once’
Future Work

- open question of equivalence for $\text{RTC}_G$, $\text{NCRTC}_G^{\omega}$ and $\text{CRTC}_G^{\omega}$

- Implementing $\text{CRTC}_G^{\omega}$ to support automated reasoning.

- Use $\text{TC}$ to better study implicit vs explicit induction.

- Adapt $\text{TC}$ for coinductive reasoning?
(Non-reflexive) transitive closure is a least fixed point

\[ R^+ = \mu X. \Psi_R(X) \quad \Psi_R(S) = R \cup (R \circ S) \]

The greatest fixed point gives the transitive co-closure

- Pairs \((s, t)\) in \(\nu X. \Psi_R(X)\) are those connected by a possibly infinite number of \(R\)-steps

- We can write \((RTC_{x,y}^{op} \varphi)(s, t)\) to denote that \((s, t)\) is in the reflexive, transitive co-closure of \(\varphi\)
We have the following standard semantics

\[
M, v \models (RTC_{x,y}^{op} \varphi)(s, t) \iff \\
\exists (\vec{a}_i)_{i \geq 0}. \forall i \geq 0. a_i = v(t) \lor M, v[x := a_i, y := a_{i+1}] \models \varphi
\]

E.g. The following formula defines possibly infinite lists

\[
(RTC_{x,y}^{op} \exists z. x = \text{cons}(z, y))(v, [ ])
\]