1 Introduction

Last time we introduced term automata for representing infinite types and gave a coinductive algorithm for equirecursive type equality. Now we indicate how to extend the algorithm to handle equirecursive subtyping. Here we take types to be finite and infinite terms over the ranked alphabet $\Sigma = \{\bot, \to, \top, 1\}$, where $\to$ is binary and $\bot$, $\top$, 1 are constants. The type $\bot$ is supposed to be a subtype of all types and the type $\top$ is supposed to be a supertype of all types.

The finite types are ordered naturally by the binary relation $\leq_{\text{FIN}}$ defined inductively by

(i) $\bot \leq_{\text{FIN}} \tau \leq_{\text{FIN}} \top$ for all finite τ;

(ii) if $\sigma' \leq_{\text{FIN}} \sigma$ and $\tau \leq_{\text{FIN}} \tau'$ then $\sigma \to \tau \leq_{\text{FIN}} \sigma' \to \tau'$.

Note that the converse of (ii) holds as well. This relation captures the natural type inclusion order in that it is covariant in the range and contravariant in the domain of a function type.

In order to handle recursive types, we need to extend the ordering $\leq_{\text{FIN}}$ to infinite types in a natural way. One natural definition involves infinite sequences of finite approximations. Here we use an equivalent and simpler definition that does not involve approximations.

The *parity* of a string $x \in \{L, R\}^*$, denoted πx, is the number mod 2 of L's in x. A string x is said to be *even* if $\pi x = 0$ and *odd* if $\pi x = 1$.

Let $\leq_0$ and $\leq_1$ be the following two partial orders on Σ:

$$\begin{array}{ccccc} \bot & \leq_0 & \to & \leq_0 & \top \\ \bot & \leq_0 & 1 & \leq_0 & \top \end{array} \qquad \begin{array}{ccccc} \top & \leq_1 & \to & \leq_1 & \bot \\ \top & \leq_1 & 1 & \leq_1 & \bot. \end{array}$$

Note that $\leq_0$ and $\leq_1$ are reverses of each other. For types σ, τ, define $\sigma \leq \tau$ if $\sigma(x) \leq_{\pi x} \tau(x)$ for all $x \in \mathsf{dom}\,\sigma \cap \mathsf{dom}\,\tau$.

One can show without much difficulty that the relation $\leq$ is a partial order on types and agrees with $\leq_{\text{FIN}}$ on the finite types. In particular, for any σ, τ, σ', τ',

(i) $\bot \leq \tau \leq \top$

(ii) $\tau \leq \bot$ if and only if $\tau = \bot$

(iii) $\top \leq \tau$ if and only if $\tau = \top$

(iv) $\sigma \to \tau \leq \sigma' \to \tau'$ if and only if $\sigma' \leq \sigma$ and $\tau \leq \tau'$.

2 An Algorithm

To decide whether $[\![\sigma]\!] \leq [\![\tau]\!]$ for two given type expressions σ and τ, we proceed as in the last lecture. We first construct the term automata M_σ and M_τ, then form their product; however, we also include one extra bit of information in the state to record the parity of the path scanned so far. This is to account for contravariance of function types in their domain.

Recall that $[\![\sigma]\!] \leq [\![\tau]\!]$ iff $[\![\sigma]\!](x) \leq_{\pi x} [\![\tau]\!](x)$ for all $x \in \mathsf{dom}\,[\![\sigma]\!] \cap \mathsf{dom}\,[\![\tau]\!]$. Equivalently, $[\![\sigma]\!] \not\leq [\![\tau]\!]$ iff the set

$$\{x \in \mathsf{dom}\,[\![\sigma]\!] \cap \mathsf{dom}\,[\![\tau]\!] \mid [\![\sigma]\!](x) \not\leq_{\pi x} [\![\tau]\!](x)\}$$

is nonempty. This is a regular subset of $\{L, R\}^*$, as it is the set accepted by the finite-state automaton

$$(Q,\, \{L, R\},\, s,\, \delta,\, F)$$

where

- $Q \triangleq Q_\sigma \times Q_\tau \times \{0, 1\}$ are the states;
- $s \triangleq (s_\sigma, s_\tau, 0)$ is the start state;
- $\delta : \{L, R\} \to Q \rightharpoonup Q$ is the partial function which for $b \in \{0, 1\}$, $D \in \{L, R\}$, $p \in Q_\sigma$, and $q \in Q_\tau$ gives

  $$\delta(D)(p, q, b) \quad \triangleq \quad (\delta_\sigma(D)(p),\, \delta_\tau(D)(q),\, b \oplus \pi D)$$

  where $\oplus$ denotes mod 2 sum;
- $F \triangleq \{(p, q, b) \mid \ell_\sigma(p) \not\leq_b \ell_\tau(q)\}$ is the set of accept states.

Then $\delta(D)(p, q, b)$ is defined if and only if $\ell_\sigma(p) = \ell_\tau(q) = \to$. The automaton is nondeterministic only in that the state (p, q, b) has no D-successors if either $\ell_\sigma(p)$ or $\ell_\tau(q) \in \{\bot, \top, 1\}$. If $\ell_\sigma(p) = \ell_\tau(q) = \to$, then the D-successor of (p, q, b) is defined and is unique.

Now to decide whether $[\![\sigma]\!] \leq [\![\tau]\!]$, we construct the automaton and ask whether it accepts a nonempty set, that is, whether there exists a path from the start state to some final state. This can be determined in linear time in the size of the automaton using depth first search.

The automaton has $2 \cdot |Q_\sigma| \cdot |Q_\tau|$ states and at most two transition edges from each state. Thus the entire algorithm takes no more than $O(|\sigma| \cdot |\tau|)$ time, where $|\sigma|$ and $|\tau|$ are the sizes of the original type expressions representing the regular terms $[\![\sigma]\!]$ and $[\![\tau]\!]$.

3 Soundness and Completeness of Type Equivalence

Last time we introduced the following proof system for type equivalence. Judgments are sequents of the form $E \vdash \sigma = \tau$, where E is a set of type equations.

$$E,\, \sigma = \tau \vdash \sigma = \tau \qquad\qquad E \vdash 1 = 1$$

$$\frac{E,\, \mu\alpha.\,\sigma = \tau \vdash \sigma\{\mu\alpha.\,\sigma/\alpha\} = \tau}{E \vdash \mu\alpha.\,\sigma = \tau} \qquad \frac{E,\, \tau = \mu\alpha.\,\sigma \vdash \tau = \sigma\{\mu\alpha.\,\sigma/\alpha\}}{E \vdash \tau = \mu\alpha.\,\sigma} \qquad \frac{E \vdash \sigma_1 = \sigma_2 \quad E \vdash \tau_1 = \tau_2}{E \vdash \sigma_1 \to \tau_1 = \sigma_2 \to \tau_2}$$

This system is slightly modified from the system presented last time in that we have gotten rid of the symmetry rule and, to compensate, introduced two versions of the fold rule. As noted last time, soundness is the main issue; even seemingly innocuous modifications, such as the introduction of a transitivity rule, break the soundness of the system.

3.1 Bisimulation

A *bisimulation* between two term automata $M = (Q_M,\, \delta_M,\, \ell_M,\, s_M)$ and $N = (Q_N,\, \delta_N,\, \ell_N,\, s_N)$ is a relation $\approx \subseteq Q_M \times Q_N$ such that

1. $s_M \approx s_N$
2. for all $u \in Q_M$ and $v \in Q_N$, if $u \approx v$, then
   (a) $\ell(u) = \ell(v)$
   (b) if $\ell_M(u) = \ell_N(v) = \to$, then for all $a \in \{L, R\}$, $\delta_M(a)(u) \approx \delta_N(a)(v)$.

The automata M and N are said to be *bisimilar* if there is a bisimulation between them.

Lemma 1. *M and N are bisimilar iff $[\![M]\!] = [\![N]\!]$.*

Proof. For the forward implication, one can show by induction on the length of $x \in \{L, R\}^*$ that if $u \approx v$, then either both $\widehat{\delta}_M(x)(u)$ and $\widehat{\delta}_N(x)(v)$ are undefined or both are defined and $\widehat{\delta}_M(x)(u) \approx \widehat{\delta}_N(x)(v)$. It follows that for all $x \in \{L, R\}^*$, $\ell_M(\widehat{\delta}_M(x)(u)) = \ell_N(\widehat{\delta}_N(x)(v))$. Since $s_M \approx s_N$, we have

$$[\![M]\!] \;=\; \lambda x \in \{L, R\}^*.\, \ell_M(\widehat{\delta}_M(x)(s_M)) \;=\; \lambda x \in \{L, R\}^*.\, \ell_N(\widehat{\delta}_N(x)(s_N)) \;=\; [\![N]\!].$$

For the converse, set $u \approx v$ if $\ell_M(\widehat{\delta}_M(x)(u)) = \ell_N(\widehat{\delta}_N(x)(v))$ for all $x \in \{L, R\}^*$. It is straightforward to verify that if $[\![M]\!] = [\![N]\!]$, then this is a bisimulation. In fact, one can show that it is the unique maximal bisimulation. □

3.2 Soundness

Now let $M_\sigma = (Q_\sigma, \ldots)$ and $M_\tau = (Q_\tau, \ldots)$ be the term automata defined from σ and τ, respectively, as defined in the last lecture.

Theorem 2 (Soundness)**.** *Suppose $\sigma = \tau$ is provable in the above deductive system. Define the relation $\approx \subseteq Q_\sigma \times Q_\tau$ by: $\pi \approx \rho$ if a judgment of the form $E \vdash \pi = \rho$ appears in the proof of $\vdash \sigma = \tau$. The relation $\approx$ is a bisimulation between M_σ and M_τ, therefore by Lemma 1, $[\![\sigma]\!] = [\![\tau]\!]$.*

Proof. Certainly $\sigma \approx \tau$, since the judgment $\vdash \sigma = \tau$ appears at the root of the proof tree, and σ and τ are the start states of M_σ and M_τ, respectively. This establishes clause 1 in the definition of bisimulation.

For clause 2(a), we proceed by induction on the well-founded relation

$$(\sigma\{\mu\alpha.\,\sigma/\alpha\}, \tau) \;<\; (\mu\alpha.\,\sigma, \tau) \qquad\qquad (\tau, \sigma\{\mu\alpha.\,\sigma/\alpha\}) \;<\; (\tau, \mu\alpha.\,\sigma)$$

The relation is well-founded by the restriction that we may not have $\mu\alpha.\,\sigma$ with σ a variable.

Suppose $\pi \approx \rho$. Then $E \vdash \pi = \rho$ appears in the proof tree. We have four cases, depending on the rule that is applied to derive $E \vdash \pi = \rho$.

If the rule is that axiom $E \vdash 1 = 1$, then $\ell(\pi) = \ell(\rho) = 1$, and if the rule is the $\to$-introduction rule, then $\ell(\pi) = \ell(\rho) = \to$.

If the rule is the left folding rule, then $\pi = \mu\alpha.\,\varphi$ and the unique premise is $E, \pi = \rho \vdash \varphi\{\pi/\alpha\} = \rho$. Then $\ell(\pi) = \ell(\varphi\{\pi/\alpha\}) = \ell(\rho)$, the first equality by the definition of ℓ and the second by the induction hypothesis on the well-founded relation $<$. The right folding rule is symmetric.

Finally, if it is the axiom $E \vdash \pi = \rho$ where $\pi = \rho \in E$, there must be an ancestor in the proof tree of the form $E' \vdash \pi = \rho$ with $E' \subseteq E - \{\pi = \rho\}$ derived by an application of one of the two folding rules (say the left) from the premise $E', \pi = \rho \vdash \varphi\{\pi/\alpha\} = \rho$, where $\pi = \mu\alpha.\,\varphi$. Then this case reverts to the previous case.

For clause 2(b), we again proceed by induction on the same well-founded relation.

Suppose $\pi \approx \rho$. Then $E \vdash \pi = \rho$ appears in the proof tree. We again have four cases, depending on the rule that is applied to derive $E \vdash \pi = \rho$.

If the rule is $E \vdash 1 = 1$, then $\delta(L)(\pi)$ and $\delta(L)(\rho)$ are undefined, and similarly for R.

If the rule is the $\to$-introduction rule, say $\pi = \pi_L \to \pi_R$ and $\rho = \rho_L \to \rho_R$ with premises $E \vdash \pi_L = \rho_L$ and $E \vdash \pi_R = \rho_R$, then $\delta(L)(\pi) = \pi_L \approx \rho_L = \delta(L)(\rho)$, and similarly for R.

If the rule is one of the folding rules, say the left one (the right is symmetric), then $\pi = \mu\alpha.\, \varphi$ and the unique premise is E, $\pi = \rho \vdash \varphi\{\pi/\alpha\} = \rho$. Then $\delta(L)(\pi) = \delta(L)(\varphi\{\pi/\alpha\}) \approx \delta(L)(\rho)$ by definition of δ and the induction hypothesis, and similarly for R.

For the axiom $E \vdash \pi = \rho$ where $\pi = \rho \in E$, the case reverts to the previous case for the same reason as above. □

3.3 Completeness

Theorem 3 (Completeness)**.** *If $[\![\sigma]\!] = [\![\tau]\!]$, then $\vdash \sigma = \tau$ is provable in the deductive system.*

Proof sketch. If $[\![\sigma]\!] = [\![\tau]\!]$, then by Lemma 1, there is a bisimulation $\approx$ between M_σ and M_τ. We can build a proof tree inductively starting at the root with label $\vdash \sigma = \tau$. Since σ and τ are the start states of M_σ and M_τ respectively, we have $\sigma \approx \tau$. Now we build the proof tree upwards by applying any rule that applies. One can show inductively that $\pi \approx \rho$ for any judgment $E \vdash \pi = \rho$ so generated, so $\ell(\pi) = \ell(\rho)$, therefore either $\pi = \rho = 1$ or the path can be extended. Continue extending every path until the path ends in a leaf of the form $E \vdash 1 = 1$ or until a repetition occurs, in which case the path can be cut off with the rule E, $\pi = \rho \vdash \pi = \rho$. □