Coinduction in Agda via Copatterns and Sized Types

Andreas Abel

Department of Computer Science and Engineering
Chalmers and Universität Göteborg

Dagstuhl Seminar 16131
Language Based Verification Tools for Functional Programs
30 March 2016

Agda's new coinduction

- Type-based termination using sized types (1996-).
- Overcome limits of syntactic termination checking.
- Workarounds for inductive case: measures, well-founded relations.
- Don't work for coinductive case (productivity checking).
- Copattern matching was invented for type-based productivity checking.
- Corecursion via copattern matching dualizes recursion via pattern matching.
- Foundations: new article Abel/Pientka JFP 2016.

Languages (infinite tries)

- $\mathsf{Lang} \cong \mathsf{Bool} \times (A \to \mathsf{Lang})$
- Coinductive tries Lang defined via observations/projections ν and δ:
- Lang is the greatest type consistent with these rules:

$$\frac{l : \mathsf{Lang}}{\nu\, l : \mathsf{Bool}} \qquad \frac{l : \mathsf{Lang} \qquad a : A}{\delta\, l\, a : \mathsf{Lang}}$$

- Empty language $\emptyset : \mathsf{Lang}$.
- Language of the empty word $\varepsilon : \mathsf{Lang}$ defined by copattern matching:

$$\begin{array}{lcl} \nu\,\varepsilon & = & \mathsf{true} \\ \delta\,\varepsilon\,a & = & \emptyset \end{array}$$

Corecursion

- Empty language $\emptyset : \mathsf{Lang}$ defined by corecursion:

$$\begin{array}{lcl} \nu\, \emptyset & = & \mathsf{false} \\ \delta\, \emptyset\, a & = & \emptyset \end{array}$$

- Language union $k \cup l$ is pointwise disjunction:

$$\begin{array}{lcl} \nu\, (k \cup l) & = & \nu\, k \vee \nu\, l \\ \delta\, (k \cup l)\, a & = & \delta\, k\, a \cup \delta\, l\, a \end{array}$$

- Language composition $k \cdot l$ à la Brzozowski:

$$\begin{array}{lcl} \nu\, (k \cdot l) & = & \nu\, k \wedge \nu\, l \\ \delta\, (k \cdot l)\, a & = & \begin{cases} (\delta\, k\, a \cdot l) \cup \delta\, l\, a & \text{if } \nu\, k \\ (\delta\, k\, a \cdot l) & \text{otherwise} \end{cases} \end{array}$$

- Not accepted because $\cup$ is not a constructor.

Sized coinductive types

- $\mathsf{Lang}\, i \cong \mathsf{Bool} \times (\forall j{<}i.\ A \to \mathsf{Lang}\, j)$

$$\frac{l : \mathsf{Lang}\, i}{\nu\, l : \mathsf{Bool}} \qquad \frac{l : \mathsf{Lang}\, i \qquad j < i \qquad a : A}{\delta\, l\, \{j\}\, a : \mathsf{Lang}\, j}$$

- $\emptyset : \forall i.\, \mathsf{Lang}\, i$ by copatterns and induction on i:

$$\begin{array}{lcl} \nu\, (\emptyset\, \{i\}) & = & \mathsf{false} \\ \delta\, (\emptyset\, \{i\})\, \{j\}\, a & = & \emptyset\, \{j\} \end{array}$$

Type-based guardedness checking

- Union preserves size/guardeness:

$$\frac{k : \mathsf{Lang}\, i \qquad l : \mathsf{Lang}\, i}{k \cup l : \mathsf{Lang}\, i}$$

$$\begin{array}{lcl} \nu\,(k \cup l) & = & \nu\, k \vee \nu\, l \\ \delta\,(k \cup l)\,\{j\}\, a & = & \delta\, k\,\{j\}\, a \cup \delta\, l\,\{j\}\, a \end{array}$$

- Composition is accepted and also guardedness-preserving:

$$\frac{k : \mathsf{Lang}\, i \qquad l : \mathsf{Lang}\, i}{k \cdot l : \mathsf{Lang}\, i}$$

$$\begin{array}{lcl} \nu\,(k \cdot l) & = & \nu\, k \wedge \nu\, l \\ \delta\,(k \cdot l)\,\{j\}\, a & = & \begin{cases} (\delta\, k\,\{j\}\, a \cdot l) \cup \delta\, l\,\{j\}\, a & \text{if } \nu\, k \\ (\delta\, k\,\{j\}\, a \cdot l) & \text{otherwise} \end{cases} \end{array}$$

Bisimilarity

- Equality of infinite tries is defined coinductively.
- $\_ \cong \_$ is the greatest relation consistent with

$$\frac{l \cong k}{\nu\, l = \nu\, k} \qquad \frac{l \cong k \qquad a : A}{\delta\, l\, a \cong \delta\, k\, a}$$

- Equivalence relation.
- Congruence for language constructions.

$$\frac{k \cong k' \qquad l \cong l'}{(k \cup k') \cong (l \cup l')}$$

- Prove language laws:

$$(k \cup l) \cdot m \cong (k \cdot m) \cup (l \cdot m)$$