A Predicative Analysis of Structural Recursion

Andreas Abel

November 19, 1999

Slide 1

- The termination checker foetus
- Soundness of structural recursion by an impredicative semantic analysis
- A predicative semantic analysis

Slide 2

Motivation (1): Division by 2

half 0 $= 0$
half 1 $= 0$
half n+2 = (half n)+1

$\mathrm{half'} = \mathsf{R}^{\mathsf{N}}\,(\lambda x^{\mathsf{B}}.\,0)\,(\lambda x^{\mathsf{N}}\lambda f^{\mathsf{B}\to\mathsf{N}}.\,\mathsf{R}^{\mathsf{B}}\,(f\,\mathsf{true})\,(1 + (f\,\mathsf{false})))$
$\mathrm{half} = \lambda n^{\mathsf{N}}.\,\mathrm{half'}\,n\,\mathsf{false}$

Slide 3

Motivation (2): Pattern matching in LEGO

```
[leRefl: {...}{v:...}vLe v v];
[...
    leRefl vUnit ==> leUnit
 || leRefl (vInl S v) ==> leInl S S (leRefl v)
 || leRefl (vInr S v) ==> leInr S S (leRefl v)
 || leRefl (vPair v w) ==> lePair (leRefl v) (leRefl w)
 || leRefl (vFold R x) ==> leFoldl R (leFoldr R (leRefl x))
];
```

Slide 4

Structural Recursion

$$\forall\, w < v.\ \mathsf{f}(w)\Downarrow\ \Rightarrow\ \mathsf{f}(v)\Downarrow$$

Example: Addition of Ordinal Numbers

```
datatype Nat = ...
datatype Ord = O
             | S of Ord
             | Lim of Nat -> Ord;
fun addord x O       = x
  | addord x (S y')  = S (addord x y')
  | addord x (Lim f) = Lim (fn z:Nat => addord x (f z))
```

Slide 5

Lexicographic Ordering

Example: Ackermann Function

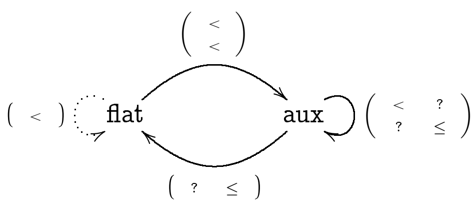
```
fun ack O     y     = S y
  | ack (S x) O     = ack x (S O)
  | ack (S x) (S y) = ack x (ack (S x) y)
```

Slide 6

Mutual Recursive Functions

Example: List Flattening

```
fun  flat []      = []
   | flat (l::ls) = aux l ls
and aux []      ls = flat ls
   | aux (x::xs) ls = x :: aux xs ls;
```



Slide 7

Soundness of structural recursion

$$f(u_0) \leadsto f(u_1) \leadsto f(u_2) \leadsto \dots$$
$$u_0 > u_1 > u_2 > \dots$$

- Define types σ, terms, values Val^{σ}
- Define semantics $[\![\sigma]\!] \subseteq \mathrm{Val}^{\sigma}$
- Define ordering $<$ on $[\![\sigma]\!]$
- Show *wellfoundedness* of $[\![\sigma]\!]$ w.r.t. $<$

Slide 8

The foetus system

	Type	Terms / **Values**	Explanation
(Var)	X, Y, Z, ...	–	type variables
(Sum)	$\Sigma(\sigma_1, \dots, \sigma_n)$	**in**$_j$, case	finite disjoint sum
(Prod)	$\Pi(\sigma_1, \dots, \sigma_n)$	(–,. . . ,–), pi$_j$	finite product
(Arr)	$\sigma \to \tau(\vec{X})$	λ, **rec**, – – (app)	function space
(Mu)	$\mu X.\sigma(X)$	**fold**, unfold	inductive type

$$\sigma(\mu X.\sigma) \underset{\text{unfold}}{\overset{\text{fold}}{\rightleftarrows}} \mu X.\sigma(X)$$

Slide 9

Operational semantics

Closures Cl^{σ}:

$\langle t^{\sigma}; e\rangle$	t term, e environment
$f^{\rho\to\sigma}@u^{\rho}$	f function value, u argument value

Evaluation relation $\Downarrow^{\sigma} \subseteq \mathrm{Cl}^{\sigma} \times \mathrm{Val}^{\sigma}$:

- big step
- call-by-value
- fixed evaluation strategy

Slide 10

Semantics

Let $\vec{V} \subseteq \mathrm{Val}^{\vec{\tau}}$. Define $[\![\sigma(\vec{X})]\!](\vec{V})$ inductively:

(Var)	$[\![X_i]\!](\vec{V}) := V_i$
(Sum)	$[\![\Sigma\vec{\sigma}]\!](\vec{V}) := \bigcup_{j=1}^{n}\{\mathrm{in}_j(v) : v \in [\![\sigma_j]\!](\vec{V})\}$
(Prod)	$[\![\Pi\vec{\sigma}]\!](\vec{V}) := \{(\vec{v}) : v_i \in [\![\sigma_i]\!](\vec{V})\}$
(Arr)	$[\![\sigma \to \tau(\vec{X})]\!](\vec{V}) := \{f \in \mathrm{Val}^{\sigma\to\tau(\vec{\tau})} : \forall u \in [\![\sigma]\!].$ $\exists v \in [\![\tau(\vec{X})]\!](\vec{V}).\ f@u \Downarrow v\}$
(Mu)	$[\![\mu Y.\, \sigma(\vec{X}, Y)]\!](\vec{V})$ smallest set closed under $\dfrac{v \in [\![\sigma]\!](\vec{V}, [\![\mu Y.\, \sigma]\!](\vec{V}))}{\mathrm{fold}(v) \in [\![\mu Y.\, \sigma]\!](\vec{V})}$ (using Knaster-Tarski)

Slide 11

Example: Lists

$$[\![\underbrace{\mu Y.\, 1 + X \times Y}_{\mathsf{List}(X)}]\!](\{a, b, c\}) \subseteq [\![\mathsf{List}(X)]\!](\mathsf{Val}^{\tau})$$

$$[\![\mathsf{List}(X)]\!]([\![\underbrace{\mu X.\, 1 + X}_{\mathsf{Nat}}]\!]) = [\![\mathsf{List}(\mathsf{Nat})]\!]$$

Slide 12

Structural Ordering

Define $<, \leq \subseteq [\![\sigma]\!] \times [\![\tau]\!]$:

- $v \leq \mathrm{in}_j(v)$
- $v_i \leq (\vec{v})$
- $f(v) \leq f$
- $v < \mathrm{fold}(v)$
- (reflexive) transitive closure

Slide 13

Wellfoundedness

Define $\mathrm{Acc}^{\sigma} \subseteq [\![\sigma]\!]$ inductively:

$$\frac{\forall w < v.\, w \in \mathrm{Acc}^{\tau}}{v \in \mathrm{Acc}^{\sigma}}$$

Theorem. $[\![\sigma(X_1, \ldots, X_n)]\!](\mathrm{Acc}^{\tau_1}, \ldots \mathrm{Acc}^{\tau_n}) \subseteq \mathrm{Acc}^{\sigma(\tau_1, \ldots, \tau_n)}$

Proof by induction on σ.

Slide 14

Lemma (acc^{-1})**.**

$$\frac{v \in \mathrm{Acc}^{\sigma} \qquad w < v}{w \in \mathrm{Acc}^{\tau}}$$

Lemma (Destructors for Acc).

$$\frac{\mathsf{in}_j(v) \in \mathrm{Acc}^{\Sigma\vec{\sigma}}}{v \in \mathrm{Acc}^{\sigma_j}} \qquad \frac{(\vec{v}) \in \mathrm{Acc}^{\Pi\vec{\sigma}}}{v_i \in \mathrm{Acc}^{\sigma_i}}$$

$$\frac{f \in \mathrm{Acc}^{\sigma \to \tau} \qquad f@u \Downarrow v}{v \in \mathrm{Acc}^{\tau}} \qquad \frac{\mathsf{fold}(v) \in \mathrm{Acc}^{\mu X.\sigma}}{v \in \mathrm{Acc}^{\sigma(\mu X.\sigma)}}$$

Proof by (acc^{-1}).

Lemma.

$$\frac{v \in \mathrm{Acc}^{\tau} \qquad w \le v}{w \in \mathrm{Acc}^{\sigma}}$$

Proof by induction on $w \le v$ using destructor property.

Slide 15

Lemma.

$$\frac{v \in \mathrm{Acc}^{\sigma_j}}{\mathrm{in}_j(v) \in \mathrm{Acc}^{\Sigma\vec{\sigma}}} \qquad \frac{v_i \in \mathrm{Acc}^{\sigma_i} \text{ for all } i}{(\vec{v}) \in \mathrm{Acc}^{\Pi\vec{\sigma}}}$$

$$\frac{f \in [\![\sigma \to \tau]\!] \qquad \forall u \in [\![\sigma]\!].\, f@u \Downarrow v \in \mathrm{Acc}^{\tau}}{f \in \mathrm{Acc}^{\sigma \to \tau}} \qquad \frac{v \in \mathrm{Acc}^{\sigma(\mu X.\sigma)}}{\mathrm{fold}(v) \in \mathrm{Acc}^{\mu X.\sigma}}$$

Proof: Show $\forall w < [-].\, w \in \mathrm{Acc}$ by case analysis on $w < [-]$.

Slide 16

Predicative semantics construction

Avoid Knaster-Tarski, use only strictly positive inductive definitions on the meta-level.

For $\sigma(\vec{X}), \vec{\tau}$ define "has urelement"-relations $\mathcal{U}_i^{\sigma} \subseteq \mathrm{Val}^{\sigma(\vec{\tau})} \times \mathrm{Val}^{\tau_i}$

with properties

$$\text{(I)} \quad \frac{v \in [\![\sigma]\!](\vec{V}) \qquad v\, \mathcal{U}_i^{\sigma}\, u}{u \in V_i}$$

$$\text{(II)} \quad \frac{v \in [\![\sigma]\!](\mathrm{Val}^{\vec{\tau}})}{v \in [\![\sigma]\!](\vec{\mathcal{U}}^{\sigma}(v))}$$

Proposition.

$$v \in [\![\sigma]\!](\vec{V}) \iff v \in [\![\sigma]\!](\mathrm{Val}^{\vec{\tau}}) \;\&\; \vec{\mathcal{U}}^{\sigma}(v) \subseteq \vec{V}$$

Slide 17

Strictly positive definition of $[\![\mu X.\,\sigma]\!]$:

$$\frac{v \in [\![\sigma]\!](\vec{V}, \mathsf{Val}^{\mu X.\sigma(\vec{\tau})}) \qquad \mathcal{U}^{\sigma}_{n+1}(v) \subseteq [\![\mu X.\,\sigma]\!](\vec{V})}{\mathsf{fold}(v) \in [\![\mu X.\,\sigma]\!](\vec{V})}$$

Task. By induction on σ

- Define $[\![\sigma]\!]$ and $\mathcal{U}^{\sigma}_{i}$.
- Verify monotonicity of $[\![\sigma]\!]$.
- Verify properties (I) and (II) of $\mathcal{U}^{\sigma}_{i}$.

Slide 18

The $\mathcal{U}$-relation for Lists

$$\mathsf{List}(X) = \mu Y.\,1 + X \times Y$$

$$\mathcal{U} \subseteq \mathsf{Val}^{\mathsf{List}(\tau)} \times \mathsf{Val}^{\tau}$$

$$\frac{}{\mathsf{fold}(\mathsf{in}_2(v, vs))\ \mathcal{U}\ v} \qquad \frac{vs\ \mathcal{U}\ u}{\mathsf{fold}(\mathsf{in}_2(v, vs))\ \mathcal{U}\ u}$$

Slide 19

Extensions and open questions

- positive types (?)
- polymorphic types $\surd$
- coinductive types
- dependent types

Soundness of foetus

- define structural recursion syntactically
- show syntactically s.r. $\Rightarrow$ semantically s.r.

Slide 20

Resources

- A. Abel, foetus – Termination Checker for Simple Functional Programs
- A. Abel, A Semantic Analysis of Structural Recursion
- A. Abel, T. Altenkirch, A Predicative Strong Normalisation Proof for a λ-calculus with Interleaving Inductive Types
- *in preparation:* A. Abel, T. Altenkirch, A Predicative Analysis of Structural Recursion