Fixed Point Calculus

Roland Backhouse

December 10, 2002

Overview

- Why a calculus?
- Equational Laws
- Application

Specification $\neq$ Implementation

Suppose Prolog is being used to model family relations. Suppose $\mathsf{parent}(X,Y)$ represents the relationship X is a parent of Y and suppose $\mathsf{ancestor}(X,Y)$ is the transitive closure of the parent relation. Then

$$\mathsf{ancestor}(X,Y) \Leftarrow \mathsf{parent}(X,Y)$$

and

$$\mathsf{ancestor}(X,Y) \Leftarrow \exists\langle Z :: \mathsf{ancestor}(X,Z) \wedge \mathsf{ancestor}(Z,Y)\rangle \quad .$$

However,

$$\begin{array}{lll} \mathsf{ancestor}(X,Y) & :- & \mathsf{parent}(X,Y) \quad . \\ \mathsf{ancestor}(X,Y) & :- & \mathsf{ancestor}(X,Z) \quad , \quad \mathsf{ancestor}(Z,Y) \quad . \end{array}$$

is not a correct Prolog implementation.

$$\begin{array}{lll} \mathsf{ancestor}(X,Y) & :- & \mathsf{parent}(X,Y) \quad . \\ \mathsf{ancestor}(X,Y) & :- & \mathsf{parent}(X,Z) \quad , \quad \mathsf{ancestor}(Z,Y) \quad . \end{array}$$

is a correct implementation.

Specification $\neq$ Implementation

The grammar

$$\langle \text{StatSeq} \rangle \quad ::= \quad \langle \text{Statement} \rangle \quad | \quad \langle \text{StatSeq} \rangle \;\; ; \;\; \langle \text{StatSeq} \rangle$$

describes a sequence of statements separated by semicolons. But it is ambiguous and not amenable to top-down or bottom-up parsing.

The grammar

$$\begin{array}{lcl} \langle \text{StatSeq} \rangle & ::= & \langle \text{Statement} \rangle \langle \text{Rest} \rangle \\ \langle \text{Rest} \rangle & ::= & \varepsilon \quad | \quad ; \; \langle \text{Statement} \rangle \; \langle \text{Rest} \rangle \end{array}$$

is equivalent and amenable to parsing by recursive descent.

The grammar

$$\langle \text{StatSeq} \rangle \quad ::= \quad \langle \text{Statement} \rangle \quad | \quad \langle \text{StatSeq} \rangle \;\; ; \;\; \langle \text{Statement} \rangle$$

is also equivalent and preferable for bottom-up parsing.

Specification $\neq$ Implementation

Testing whether the empty word is generated by a grammar is easy.
For example, given the grammar

$$S \quad ::= \quad \varepsilon \mid aS$$

we construct and solve the equation

$$\varepsilon{\in}S \quad = \quad \varepsilon{\in}\{\varepsilon\} \vee (\varepsilon{\in}\{a\} \wedge \varepsilon{\in}S)$$

But it is not the case that (eg)

$$a{\in}S \quad = \quad a{\in}\{\varepsilon\} \vee (a{\in}\{a\} \wedge a{\in}S)$$

(The least solution is $a{\in}S = \mathsf{false}$.)

The general membership test is a non-trivial problem!

Least Fixed Points

Recall the characterising properties of least fixed points:

computation rule

$$\mu f \;=\; f.\mu f$$

induction rule: for all $x{\in}\mathcal{A}$,

$$\mu f \leq x \quad\Leftarrow\quad f.x \leq x \quad .$$

The induction rule is undesirable because it leads to proofs by mutual inclusion (i.e. the consideration of two separate cases).

Closure Rules

In any Kleene algebra

$$a^* \;=\; \langle \mu x :: 1 + x \cdot a \rangle \;=\; \langle \mu x :: 1 + a \cdot x \rangle \;=\; \langle \mu x :: 1 + a + x \cdot x \rangle$$

$$a^+ \;=\; \langle \mu x :: a + x \cdot a \rangle \;=\; \langle \mu x :: a + a \cdot x \rangle \;=\; \langle \mu x :: a + x \cdot x \rangle$$

Basic Rules

The *rolling rule*:

$$\mu(f \circ g) \;=\; f.\mu(g \circ f) \quad . \tag{1}$$

The *square rule*:

$$\mu f \;=\; \mu(f^2) \quad . \tag{2}$$

The *diagonal rule*:

$$\langle \mu x :: x \oplus x \rangle \quad = \quad \langle \mu x :: \langle \mu y :: x \oplus y \rangle \rangle \quad . \tag{3}$$

Examples

$$\langle \mu X :: a \cdot X^* \rangle \;=\; a^+ \quad .$$

$$\langle \mu X :: a + X \cdot b \cdot X \rangle \;=\; a \cdot (b \cdot a)^* \quad .$$

Fusion

Many problems are expressed in the form

$$evaluate \circ generate$$

where $generate$ generates a (possibly infinite) candidate set of solutions, and $evaluate$ selects a best solution.

Examples:

$$shortest \circ path \ ,$$

$$(x\in) \circ L \ .$$

Solution method is to *fuse* the generation and evaluation processes, eliminating the need to generate all candidate solutions.

Language Problems

$$S \ ::= \ aSS \mid \varepsilon \ .$$

Is-empty

$$S=\phi \quad \equiv \quad (\{a\}=\phi \vee S=\phi \vee S=\phi) \wedge \{\varepsilon\}=\phi \ .$$

Nullable

$$\varepsilon \in S \quad \equiv \quad (\varepsilon \in \{a\} \wedge \varepsilon \in S \wedge \varepsilon \in S) \vee \varepsilon \in \{\varepsilon\} \ .$$

Shortest word length

$$\#S \ = \ (\#a + \#S + \#S) \downarrow \#\varepsilon \ .$$

Non-Example

$$aa \in S \quad \not\equiv \quad (aa \in \{a\} \wedge aa \in S \wedge aa \in S) \vee aa \in \{\varepsilon\} \ .$$

Conditions for Fusion

Fusion is made possible when

- $\mathit{evaluate}$ is an adjoint in a *Galois connection*,
- $\mathit{generate}$ is expressed as a *fixed point*.

Fusion Theorem

$$F.(\mu_{\preceq}g) \quad = \quad \mu_{\sqsubseteq}h$$

provided that

- F is a lower adjoint in a Galois connection of $\sqsubseteq$ and $\preceq$ (see brief summary of definition below)
- $F \circ g \quad = \quad h \circ F$.

Galois Connection

$$F.x \sqsubseteq y \quad \equiv \quad x \preceq G.y \quad .$$

F is called the *lower* adjoint and G the *upper* adjoint.

Shortest Word Problem

Given a language L defined by a context-free grammar, determine the length of the shortest word in the language.

For concreteness, use the grammar

$$S \quad ::= \quad aS \quad | \quad SS \quad | \quad \varepsilon \quad .$$

The language defined by this grammar is

$$\langle \mu X :: \{a\} \cdot X \cup X \cdot X \cup \{\varepsilon\} \rangle \quad .$$

Now, for arbitrary language L,

$$\#L \;=\; \langle \Downarrow w : w \in L : \mathsf{length}.w \rangle$$

and we are required to determine

$$\# \langle \mu X :: \{a\} \cdot X \cup X \cdot X \cup \{\varepsilon\} \rangle \quad .$$

Shortest Word Problem (Continued)

For arbitrary language L,

$$\#L \;=\; \langle \Downarrow w : w \in L : \mathsf{length}.w \rangle$$

and we are required to determine

$$\# \langle \mu X :: \{a\} \cdot X \cup X \cdot X \cup \{\varepsilon\} \rangle \quad .$$

Because $\#$ is the infimum of the length function it is the lower adjoint in a Galois connection. Indeed,

$$\#L \geq k \;\;\equiv\;\; L \subseteq \Sigma^{\geq\, k}$$

where $\Sigma^{\geq\, k}$ is the set of all words (in the alphabet Σ) whose length is at least k.

So, by fusion, for all functions f and g,

$$\#(\mu_{\subseteq} f) = \mu_{\geq} g \quad \Leftarrow \quad \# \circ f \;=\; g \circ \# \quad .$$

Applying this to our example grammar, we fill in f and calculate g so that:

$$\# \circ \langle X :: \{a\} \cdot X \cup X \cdot X \cup \{\varepsilon\} \rangle \;\;=\;\; g \circ \# \quad .$$

Shortest Word Problem (Continued)

$$
\begin{array}{ll}
 & \# \circ \langle X :: \{a\}\cdot X \cup X\cdot X \cup \{\varepsilon\} \rangle \;=\; g \circ \# \\
= & \{ \quad \text{definition of composition} \quad \} \\
 & \langle \forall X :: \#(\{a\}\cdot X \cup X\cdot X \cup \{\varepsilon\}) \;=\; g.(\# X) \rangle \\
= & \{ \quad \# \text{ is a lower adjoint and so distributes over } \cup, \\
 & \quad\;\; \text{definition of } \# \quad \} \\
 & \langle \forall X :: \#(\{a\}\cdot X) \downarrow \#(X\cdot X) \downarrow \#\{\varepsilon\} \;=\; g.(\# X) \rangle \\
= & \{ \quad \#(Y\cdot Z) = \# Y + \# Z,\; \#\{a\} = 1,\; \#\{\varepsilon\} = 0 \quad \} \\
 & (1 + \# X) \downarrow (\# X + \# X) \downarrow 0 \;=\; g.(\# X) \\
\Leftarrow & \{ \quad \text{instantiation} \quad \} \\
 & \langle \forall k :: (1+k) \downarrow (k+k) \downarrow 0 \;=\; g.k \rangle \quad .
\end{array}
$$

We conclude that

$$\# \langle \mu X :: \{a\}\cdot X \cup X\cdot X \cup \{\varepsilon\} \rangle \;=\; \langle \mu k :: (1+k) \downarrow (k+k) \downarrow 0 \rangle \quad .$$

Language Recognition

Problem: For given word x and grammar G, determine $x \in L(G)$.
That is, implement

$$(x\in) \quad \circ \quad L \quad .$$

Language $L(G)$ is the least fixed point (with respect to the subset relation) of a monotonic function.

$(x\in)$ is the lower adjoint in a Galois connection of languages (ordered by the subset relation) and booleans (ordered by implication).
(Recall,

$$x{\in}S \Rightarrow b \quad \equiv \quad S \subseteq \text{if } b \rightarrow \Sigma^* \; \square \; \neg b \rightarrow \Sigma^* - \{x\} \text{ fi} \quad .)$$

Nullable Languages

Problem: For given grammar G, determine $\varepsilon \in L(G)$.

$$(\varepsilon \in) \quad \circ \quad L$$

Solution: Easily expressed as a fixed point computation.

Works because:

- The function $(x\in)$ is a lower adjoint in a Galois connection (for all x, but in particular for $x = \varepsilon$).
- For all languages S and T,

$$\varepsilon \in S \cdot T \quad \equiv \quad \varepsilon \in S \;\wedge\; \varepsilon \in T \quad .$$

Problem Generalisation

Problem: For given grammar G, determine whether all words in $\mathsf{L}(\mathsf{G})$ have even length. I.e. implement

$$\mathsf{alleven} \;\circ\; \mathsf{L}\;.$$

The function $\mathsf{alleven}$ is a lower adjoint in a Galois connection. Specifically, for all languages S and T,

$$\mathsf{alleven}(\mathsf{S}) \Leftarrow \mathsf{b} \quad\equiv\quad \mathsf{S} \subseteq \mathsf{if}\ \neg\mathsf{b} \longrightarrow \Sigma^* \;\square\; \mathsf{b} \longrightarrow (\Sigma{\cdot}\Sigma)^*\ \mathsf{fi}$$

Nevertheless, fusion *doesn't* work (directly) because

- there is no $\otimes$ such that, for all languages S and T,

$$\mathsf{alleven}(\mathsf{S}{\cdot}\mathsf{T}) \quad\equiv\quad \mathsf{alleven}(\mathsf{S}) \otimes \mathsf{alleven}(\mathsf{T})\;.$$

Solution: Generalise by tupling: compute simultaneously $\mathsf{alleven}$ and allodd.

General Context-Free Parsing

Problem: For given grammar G, determine $x \in L(G)$.

$$(x \in) \quad \circ \quad L$$

Not (in general) expressible as a fixed point computation.

Fusion *fails* because: for all x, $x \neq \varepsilon$, there is no $\otimes$ such that, for all languages S and T,

$$x \in S{\cdot}T \qquad \equiv \qquad (x \in S) \; \otimes \; (x \in T) \quad .$$

CYK: Let $F(S)$ denote the relation $\langle i,j{::}\;\; x[i..j) \in S \rangle$.

Works because:

- The function F is a lower adjoint.
- For all languages S and T,

  $$F(S{\cdot}T) \quad = \quad F(S) \bullet F(T)$$

  where $B{\bullet}C$ denotes the composition of relations B and C.