An Albebraic Approach to Internet Routing Lectures 15 — 16

Timothy G. Griffin

timothy.griffin@cl.cam.ac.uk
Computer Laboratory
University of Cambridge, UK

Michaelmas Term
2009

Defining and implementing a new routing protocol is difficult!

- The space is large
- The proofs are difficult
- Correctness conditions hard to get right

Could the design process be partially automated?

A simple grammar for a mini-metalanguage

Our mini-metalanguage will describe routing algebras

- $(S, \oplus, F \subseteq S \rightarrow S)$
- $\oplus$ is commutative, idempotent, and has identity α.

$$
\begin{array}{rcl}
\textit{base} & ::= & \texttt{sp} \\
 & \mid & \texttt{bw} \\
 & \mid & \texttt{rel} \\
 & & \\
\textit{algebra} & ::= & \textit{term} \\
 & \mid & \texttt{right}\ \textit{term} \\
 & \mid & \texttt{left}\ \textit{term} \\
 & \mid & \texttt{lex\_product}\ \textit{term} \ldots \textit{term} \\
 & \mid & \texttt{function\_union}\ \textit{term} \ldots \textit{term} \\
 & & \\
\textit{term} & ::= & \textit{base} \\
 & \mid & (\textit{algebra})
\end{array}
$$

The Semantics

For category *base*

- $[\![\texttt{sp}]\!]^{\mathcal{B}} = (\mathbb{N} \cup \{\infty\},\ \min,\ F_{+})$
- $[\![\texttt{bw}]\!]^{\mathcal{B}} = (\mathbb{N} \cup \{\infty\},\ \max,\ F_{\min})$
- $[\![\texttt{rel}]\!]^{\mathcal{B}} = ([0,\ 1], \max,\ F_{\times})$

For category *term*

- $[\![\mathrm{b}]\!]^{\mathcal{T}} = [\![\mathrm{b}]\!]^{\mathcal{B}}$
- $[\![(\mathrm{a})]\!]^{\mathcal{T}} = [\![\mathrm{a}]\!]^{\mathcal{A}}$

The Semantics

For category *algebra*

- $[\![\texttt{t}]\!]^{\mathcal{A}} = [\![\texttt{t}]\!]^{\mathcal{B}}$
- $[\![\texttt{right t}]\!]^{\mathcal{A}} = (S,\ \oplus,\ \{i\})$
  - where $[\![\texttt{t}]\!]^{\mathcal{T}} = (S,\ \oplus,\ F)$
- $[\![\texttt{left t}]\!]^{\mathcal{A}} = (S,\ \oplus,\ K(S))$
  - where $[\![\texttt{t}]\!]^{\mathcal{T}} = (S,\ \oplus,\ F)$

The Semantics

- $[\![\texttt{lex\_product t}]\!]^{\mathcal{A}} = [\![\texttt{t}]\!]^{\mathcal{T}}$
- $[\![\texttt{lex\_product t t}']\!]^{\mathcal{A}} = (S,\ \oplus,\ F)\ \vec{\times}\ (T,\ \odot,\ G) = (S \times T,\ \oplus\ \vec{\times}\ \odot,\ F \times G)$
  - where $[\![\texttt{t}]\!]^{\mathcal{T}} = (S,\ \oplus,\ F)$
  - and $[\![\texttt{t}']\!]^{\mathcal{T}} = (T, \odot\ G)$
- $[\![\texttt{lex\_product t t}' \ldots \texttt{t}'']\!]^{\mathcal{A}} = (S,\ \oplus,\ F)\ \vec{\times}\ (T,\ \odot,\ G) = (S \times T,\ \oplus\ \vec{\times}\ \odot,\ F \times G)$
  - where $[\![\texttt{t}]\!]^{\mathcal{T}} = (S,\ \oplus,\ F)$
  - and $[\![\texttt{lex\_product t}' \ldots \texttt{t}'']\!]^{\mathcal{A}} = (T, \odot\ G)$

The Semantics

- $[\![\texttt{function\_union t}]\!]^{\mathcal{A}} = [\![\texttt{t}]\!]^{\mathcal{T}}$
- $[\![\texttt{function\_union t t}']\!]^{\mathcal{A}} = (S,\ \oplus,\ F) +_{\mathrm{m}} (S,\ \oplus,\ G) = (S,\ \oplus,\ F \cup G)$
  - where $[\![\texttt{t}]\!]^{\mathcal{T}} = (S,\ \oplus,\ F)$
  - and $[\![\texttt{t}']\!]^{\mathcal{T}} = (S, \oplus\ G)$
- $[\![\texttt{functon\_union t t}' \ldots \texttt{t}'']\!]^{\mathcal{A}} = (S,\ \oplus,\ F) +_{\mathrm{m}} (S,\ \oplus,\ G) = (S,\ \oplus,\ F \cup G)$
  - where $[\![\texttt{t}]\!]^{\mathcal{T}} = (S,\ \oplus,\ F)$
  - and $[\![\texttt{functon\_union t}' \ldots \texttt{t}'']\!]^{\mathcal{A}} = (S, \oplus\ G)$

Some interesting properties

Property	Definition
M	$\forall a, b \in S\ \forall f \in F : f(a \oplus b) = f(a) \oplus f(b)$
C	$\forall a, b \in S\ \forall f \in F - \{\omega\} : f(a) = f(b) \implies a = b$
K	$\forall a, b \in S\ \forall f \in F : f(a) = f(b)$
I	$\forall a \in S\ \forall f \in F : a \neq \alpha \implies a <_{\oplus}^{\mathrm{L}} f(a)$
ND	$\forall a \in S\ \forall f \in F : a \leq_{\oplus}^{\mathrm{L}} f(a)$

We know a few rules ...

(some of the) rules needed for global optimality

$$
\begin{aligned}
\textsc{m}(\mathbf{right}(S)) & \\
\textsc{m}(\mathbf{left}(S)) & \\
\textsc{c}(\mathbf{right}(S)) & \\
\textsc{k}(\mathbf{left}(S))\textsc{m}(S \vec{\times} T) & \iff \textsc{m}(S) \wedge \textsc{m}(T) \wedge (\textsc{c}(S) \vee \textsc{k}(T)) \\
\textsc{m}(S +_{\mathrm{m}} T) & \iff \textsc{m}(S) \wedge \textsc{m}(T)
\end{aligned}
$$

... and a few more rules

(some of the) rules needed for local optimality (and for loop-freedom in next-hop forwarding)

$$
\begin{aligned}
\textsc{i}(S \vec{\times} T) & \iff \textsc{i}(S) \vee (\textsc{nd}(S) \wedge \textsc{i}(T)) \\
\textsc{nd}(S \vec{\times} T) & \iff \textsc{i}(S) \vee (\textsc{nd}(S) \wedge \textsc{nd}(T)) \\
\textsc{i}(S +_{\mathrm{m}} T) & \iff \textsc{i}(S) \wedge \textsc{i}(T) \\
\textsc{nd}(S +_{\mathrm{m}} T) & \iff \textsc{nd}(S) \wedge \textsc{nd}(T)
\end{aligned}
$$

We can turn rules into bottom-up methods

Example : The $\iff$ rule

$$\text{M}(S \vec{\times} T) \iff \text{M}(S) \wedge \text{M}(T) \wedge (\text{C}(S) \vee \text{K}(T))$$

becomes a bottom-up method for deriving property M or property ¬M for any expression

$$e = \text{lex\_product } t_1 \ t_2$$

if derive properties for t_1	and derive properties for t_2	then derive property for e
M, C	M	M
M	M, K	M
¬M		¬M
	¬M	¬M
¬C	¬K	¬M

Magic

We know everything about our base algebras

	M	C	K	I	ND
sp	yes	yes	no	yes	yes
bw	yes	no	no	no	yes
rel	yes	yes	no	no	yes

Now, for each algebra expression a defined by our mini-metalanguage and each property P, we can determine in a bottom-up manner whether

$$\text{P}([\![\text{a}]\!]^{\mathcal{A}})$$

or

$$\neg\text{P}([\![\text{a}]\!]^{\mathcal{A}})$$

holds.

No proofs required at algebra specification time!

A few examples

	M	C	K	I	ND
`lex_product sp bw`	yes	no	no	yes	yes
`lex_product sp sp`	yes	yes	no	yes	yes
`lex_product bw sp`	no	no	no	yes	yes
`lex_product rel bw`	yes	no	no	no	yes
`lex_product rel bw sp`	yes	no	no	yes	yes

BGP-like Example

```
function_union
<internal:
    lex_product
    < ecomm: right cpp,
      epath: right paths,
      idist: sp,
      ipath: paths >,
  external:
    lex_product
    < ecomm: cpp,
      epath: paths,
      idist: left sp,
      ipath: left paths > >
```

Extracted "Code"

$$
\begin{array}{ll}
 & \text{import on internal} \\
 & \mathrm{inl}(\bot,\ \bot,\ v,\ (i,\ j)) \rhd (ec,\ ep,\ d,\ ip) \\
= & (ec,\ ep,\ v + d,\ (i,\ j) \rhd_{paths} l)
\end{array}
$$

$$
\begin{array}{ll}
 & \text{import on external} \\
 & \mathrm{inr}(x,\ (m,\ n),\ v, l) \rhd (ec,\ ep,\ d,\ ip) \\
= & (x \rhd_{cpp} ec,\ (m,\ n) \rhd_{paths} ep,\ v,\ l)
\end{array}
$$