Kleene Algebra — Lecture 2

ESSLLI 2023

Last lecture

- We discussed rational expressions and their semantics.
- We introduced a set of sound laws, and used those in proofs.
- We studied the *relational* and *language models*.

Today's lecture

- Focus on "filter programs" — the ones that "do nothing or crash".
- They admit their own equations, useful for reasoning about programs.
- Extend syntax and reasoning to obtain *Kleene Algebra with Tests*.

Filter programs

Suppose $\mathtt{a}, \mathtt{b} \in \Sigma$ are "filtering" programs, i.e.:

$$\sigma(\mathtt{a}) = \{\langle s, s \rangle \in S \times S : \phi(s)\} \qquad \sigma(\mathtt{b}) = \{\langle s, s \rangle \in S \times S : \psi(s)\}$$

In that case, you can show

$$[\![\mathtt{a} \cdot \mathtt{b}]\!]_\sigma = \{\langle s, s \rangle \in S \times S : \phi(s) \wedge \psi(s)\} = [\![\mathtt{b} \cdot \mathtt{a}]\!]_\sigma$$

Filtering programs admit more of these useful specialized equalities.

Extended syntax

We fix a set $T = \{\mathtt{t}, \mathtt{s}, \ldots\}$ of *primitive tests*.

Definition (Boolean expressions)

We write $\mathbb{B}$ for the set of *Boolean expressions*, generated by

$$\mathbb{B} \ni b, c ::= 0 \mid 1 \mid \mathtt{t} \in T \mid b + c \mid b \cdot c \mid \overline{b}$$

Definition (Guarded rational expressions)

We write $\mathbb{G}$ for the set of *guarded rational expressions*, generated by

$$\mathbb{G} \ni e, f ::= b \in \mathbb{B} \mid \mathtt{a} \in \Sigma \mid e + f \mid e \cdot f \mid e^*$$

Extended semantics — guarded interpretation

Definition (Guarded interpretation)

A *(guarded) interpretation* is a triple $\langle S, \tau, \sigma \rangle$, where

- $\langle S, \sigma \rangle$ is an interpretation; and
- $\tau : T \to 2^S$ is a function.

Intuitively: $\tau(\mathtt{t})$ is the set of states where $\mathtt{t}$ holds.

Extended semantics — Boolean expressions

Definition (Boolean expression semantics)

Let $\langle S, \tau, \sigma \rangle$ be a guarded interpretation. We define $(\!|-|\!)_\sigma : \mathbb{B} \to 2^S$ inductively:

$$(\!|0|\!)_\tau = \emptyset \qquad (\!|1|\!)_\tau = S \qquad (\!|\mathtt{t}|\!)_\tau = \tau(\mathtt{t})$$

$$(\!|b + c|\!)_\tau = (\!|b|\!)_\tau \cup (\!|c|\!)_\tau \qquad (\!|b \cdot c|\!)_\tau = (\!|b|\!)_\tau \cap (\!|c|\!)_\tau \qquad (\!|\overline{b}|\!)_\tau = S \setminus (\!|b|\!)_\tau$$

Extended semantics — guarded rational expressions

Definition (Guarded rational expression semantics)

Let $\langle S, \tau, \sigma \rangle$ be a guarded interpretation. We define $[\![-]\!]_{\sigma,\tau} : \mathbb{G} \to 2^{S \times S}$ inductively:

$$[\![b]\!]_{\sigma,\tau} = \{\langle s, s \rangle : s \in (\!|b|\!)_\tau\} \qquad [\![\mathtt{a}]\!]_{\sigma,\tau} = \sigma(\mathtt{a})$$

$$[\![e + f]\!]_{\sigma,\tau} = [\![e]\!]_{\sigma,\tau} \cup [\![f]\!]_{\sigma,\tau} \qquad [\![e \cdot f]\!]_{\sigma,\tau} = [\![e]\!]_{\sigma,\tau} \circ [\![f]\!]_{\sigma,\tau}$$

$$[\![e^*]\!]_{\sigma,\tau} = [\![e]\!]^*_{\sigma,\tau}$$

Integer square root, redux

Consider our integer square root finding program from the last lecture:

$i \leftarrow 0;$
$\texttt{while } (i+1)^2 \leq n \texttt{ do}$
$\quad i \leftarrow i+1;$

To encode this program, we had to encode both the loop guard and its negation:

$$\sigma(\texttt{guard}) = \{\langle s, s\rangle : (s(i)+1)^2 \leq s(n)\}$$

$$\sigma(\texttt{validate}) = \{\langle s, s\rangle : (s(i)+1)^2 > s(n)\}$$

Integer square root, redux

New, shiny encoding:

$$\texttt{init} \cdot (\texttt{bound} \cdot \texttt{incr})^* \cdot \overline{\texttt{bound}}$$

Let $T = \{\texttt{bound}\}$ and $\Sigma = \{\texttt{init}, \texttt{incr}\}$, and set

$$\tau(\texttt{bound}) = \{s \in S : (s(i) + 1)^2 \leq s(n)\}$$

$$\sigma(\texttt{init}) = \{\langle s, s[0/i]\rangle : s \in S\}$$

$$\sigma(\texttt{incr}) = \{\langle s, s[s(i) + 1/i]\rangle : s \in S\}$$

Encoding flow control

We can encode traditional flow control:

$$\textbf{if } b \textbf{ then } e \textbf{ else } f := b \cdot e + \overline{b} \cdot f$$

$$\textbf{if } b \textbf{ then } e := b \cdot e + \overline{b}$$

$$\textbf{while } b \textbf{ do } e := (b \cdot e)^* \cdot \overline{b}$$

This means our program can be written as

$$\texttt{init} \cdot \textbf{while } \texttt{bound} \textbf{ do } \texttt{incr}$$

Boolean equivalence — laws

Definition (Boolean algebra)

$\equiv_{\mathbb{B}}$ is the smallest congruence on $\mathbb{B}$ satisfying, for all $b, c, d \in \mathbb{B}$:

$$b + 0 \equiv_{\mathbb{B}} b \qquad b + c \equiv_{\mathbb{B}} c + b \qquad b + \overline{b} \equiv_{\mathbb{B}} 1 \qquad b + (c + d) \equiv_{\mathbb{B}} (b + c) + d$$

$$b \cdot 1 \equiv_{\mathbb{B}} b \qquad b \cdot c \equiv_{\mathbb{B}} c \cdot b \qquad b \cdot \overline{b} \equiv_{\mathbb{B}} 0 \qquad b \cdot (c \cdot d) \equiv_{\mathbb{B}} (b \cdot c) \cdot d$$

$$b + c \cdot d \equiv_{\mathbb{B}} (b + c) \cdot (b + d) \qquad b \cdot (c + d) \equiv_{\mathbb{B}} b \cdot c + b \cdot d$$

Boolean equivalence — soundness

Lemma (Soundness for Boolean algebra)

Let $b, c \in \mathbb{B}$, and let $\tau : T \to 2^S$. If $b \equiv_{\mathbb{B}} c$, then $(\!|b|\!)_\tau = (\!|c|\!)_\tau$.

Boolean equivalence — reasoning

Lemma (Opposites)

If $b, c \in \mathbb{B}$ *such that* $b + c \equiv_{\mathbb{B}} 1$ *and* $b \cdot c \equiv_{\mathbb{B}} 0$*, then* $b \equiv_{\mathbb{B}} \overline{c}$*.*

Boolean equivalence — reasoning

Lemma (DeMorgan's first law)

If $b, c \in \mathbb{B}$*, then* $\overline{b + c} \equiv \overline{b} \cdot \overline{c}$*.*

Guarded rational equivalence — laws

Definition (Kleene Algebra with Tests)

$\equiv_{\mathbb{G}}$ sas the smallest congruence on $\mathbb{G}$ satisfying, for all $b, c \in \mathbb{B}$ and $e, f, g \in \mathbb{G}$:

$$b \equiv_{\mathbb{B}} c \implies b \equiv_{\mathbb{G}} c \qquad e + 0 \equiv_{\mathbb{G}} e \qquad e + e \equiv_{\mathbb{G}} e \qquad e + f \equiv_{\mathbb{G}} f + e$$

$$e + (f + g) \equiv_{\mathbb{G}} (e + f) + g \qquad e \cdot (f \cdot g) \equiv_{\mathbb{G}} (e \cdot f) \cdot g$$

$$e \cdot (f + g) \equiv_{\mathbb{G}} e \cdot f + e \cdot g \qquad (e + f) \cdot g \equiv_{\mathbb{G}} e \cdot g + f \cdot g$$

$$e \cdot 1 \equiv_{\mathbb{G}} e \equiv_{\mathbb{G}} 1 \cdot e \qquad e \cdot 0 \equiv_{\mathbb{G}} 0 \equiv_{\mathbb{G}} 0 \cdot e \qquad 1 + e \cdot e^* \equiv_{\mathbb{G}} e^* \equiv_{\mathbb{G}} 1 + e^* \cdot e$$

$$e + f \cdot g \leqq_{\mathbb{G}} g \implies f^* \cdot e \leqq_{\mathbb{G}} g \qquad e + f \cdot g \leqq_{\mathbb{G}} f \implies e \cdot g^* \leqq_{\mathbb{G}} f$$

Guarded rational equivalence — soundness

Lemma (Soundness for Kleene Algebra with Tests)

Let $e, f \in \mathbb{G}$*, and let* $\langle S, \tau, \sigma \rangle$ *be a guarded interpretation.*

If $e \equiv_{\mathbb{G}} f$*, then* $[\![e]\!]_{\sigma,\tau} = [\![f]\!]_{\sigma,\tau}$*.*

Guarded rational equivalence — reasoning

Lemma (Branch swapping)

Let $e, f \in \mathbb{G}$ as well as $b \in \mathbb{B}$. The following holds:

$$\textbf{if } b \textbf{ then } e \textbf{ else } f \equiv \textbf{if } \overline{b} \textbf{ then } f \textbf{ else } e$$

Proof.

This is a matter of unrolling the syntactic sugar and applying our rules:

$$\begin{aligned}
\textbf{if } b \textbf{ then } e \textbf{ else } f &= b \cdot e + \overline{b} \cdot f && \text{(by definition)} \\
&\equiv \overline{b} \cdot f + b \cdot e && \text{(commutativity)} \\
&\equiv \overline{b} \cdot f + \overline{\overline{b}} \cdot e && \text{(see homework)} \\
&= \textbf{if } \overline{b} \textbf{ then } f \textbf{ else } e && \text{(by definition)} \quad \square
\end{aligned}$$

Guarded rational equivalence — reasoning

Lemma (Loop unrolling)

Let $e \in \mathbb{G}$ and $b \in \mathbb{B}$. The following holds:

$$\textbf{while } b \textbf{ do } e \equiv \textbf{if } b \textbf{ then } (e \cdot \textbf{while } b \textbf{ do } e)$$

Proof.

We again unfold the syntactic sugar and apply our rules, as follows:

$$\begin{aligned}
\textbf{while } b \textbf{ do } e &= (b \cdot e)^* \cdot \overline{b} && \text{(by definition)} \\
&\equiv (b \cdot e \cdot (b \cdot e)^* + 1) \cdot \overline{b} && \text{(unrolling)} \\
&\equiv b \cdot e \cdot (b \cdot e)^* \cdot \overline{b} + \overline{b} && \text{(distributivity)} \\
&= \textbf{if } b \textbf{ then } (e \cdot \textbf{while } b \textbf{ do } e) && \text{(by definition)} \quad \square
\end{aligned}$$

Guarded rational equivalence — reasoning

You now have the tools to prove the equivalence claimed earlier:

```
while a and b do
  e;
while a do
  f;
  while a and b do
    e;
```

$\equiv$

```
while a do
  if b then
    e;
  else
    f;
```

This is part of today's exercises.

Guarded language semantics

We can abstract from guarded interpretations using *guarded languages*.

Idea: record the order of actions, *and the tests that hold in-between*.

Definition (Guarded languages)

A *guarded word* is a word over $(2^T \cup \Sigma)^*$ of the form

$$\alpha_1 \mathtt{a}_1 \alpha_2 \mathtt{a}_2 \alpha_3 \cdots \alpha_{n-1} \mathtt{a}_{n-1} \alpha_n$$

A guarded language is a set of guarded words.

We write $(\Sigma, T)^*$ for the set of guarded languages.

Guarded language semantics

Definition (Guarded language composition)

Let $L, K \subseteq (\Sigma, T)^*$. We write $L \diamond K$ for the *guarded product*:

$$L \diamond K = \{w\alpha x : \alpha \in 2^T, w\alpha \in L, \alpha x \in K\}$$

We write $L^{(\diamond)}$ for the *guarded star*:

$$L^{(\diamond)} = 2^T \cup L \cup L \diamond L \cup L \diamond L \diamond L \cup \cdots$$

Guarded language semantics

Definition (Guarded language semantics)

We define $(\!|-|\!)_{\mathbb{G}} : \mathbb{B} \to 2^{2^T}$ inductively, as follows:

$$(\!|0|\!)_{\mathbb{G}} = \emptyset \qquad (\!|1|\!)_{\mathbb{G}} = 2^T \qquad (\!|\mathtt{t}|\!)_{\mathbb{G}} = \{\alpha \in 2^T : \mathtt{t} \in \alpha\}$$

$$(\!|b + c|\!)_{\mathbb{G}} = (\!|b|\!)_{\mathbb{G}} \cup (\!|c|\!)_{\mathbb{G}} \qquad (\!|b \cdot c|\!)_{\mathbb{G}} = (\!|b|\!)_{\mathbb{G}} \cap (\!|c|\!)_{\mathbb{G}} \qquad (\!|\overline{b}|\!)_{\mathbb{G}} = S \setminus (\!|b|\!)_{\mathbb{G}}$$

Next, we define $[\![-]\!]_{\mathbb{G}} : \mathbb{G} \to (\Sigma, T)^*$ inductively, as follows:

$$[\![b]\!]_{\mathbb{G}} = (\!|b|\!)_{\mathbb{G}} \qquad [\![\mathtt{a}]\!]_{\mathbb{G}} = \{\mathtt{a}\}$$

$$[\![e + f]\!]_{\mathbb{G}} = [\![e]\!]_{\mathbb{G}} \cup [\![f]\!]_{\mathbb{G}} \qquad [\![e \cdot f]\!]_{\mathbb{G}} = [\![e]\!]_{\mathbb{G}} \diamond [\![f]\!]_{\mathbb{G}} \qquad [\![e^*]\!]_{\mathbb{G}} = [\![e]\!]_{\mathbb{G}}^{(\diamond)}$$

Model equivalence

Theorem (Equivalence of guarded models)

Let $e, f \in \mathbb{G}$. The following are equivalent:

(i) $[\![e]\!]_{\mathbb{G}} = [\![f]\!]_{\mathbb{G}}$

(ii) *for all* σ *and* τ, $[\![e]\!]_{\sigma,\tau} = [\![f]\!]_{\sigma,\tau}$.

Proof.

Like in the last lecture, but with more Greek letters! □

From now on

- No more guarded expressions!
- Everything still works when you add tests.
- The proofs just become more involved.

Next lecture

- Representing languages using *automata*.
- Checking language equivalence of automata.
- Converting expressions to automata.