KLEENE ALGEBRA WITH TESTS

LECTURE 3

YESTERDAY: KA(T)

TODAY: LŠ SEMANTICS
LŠ EXAMPLES
- NETKAT (SYNTAX + EXAMPLES)

LA:
- DENOTATIONAL SEMANTICS WITH REGULAR LANGUAGES
  - ALLOWS REASONING SYNTACTICALLY
- OPERATIONAL SEMANTICS WITH DFAs
  - ALLOWS REASONING MECHANICALLY

KAT SYNTAX:

PROGRAM:

\[ e ::= 0 \mid \text{p} \mid \text{e} + \text{e} \mid \text{e} \cdot \text{e} \mid \text{e}^* \mid b \in \mathcal{B} \]

BOOLEAN:

\[ b ::= \text{b} \mid \text{b} \mid \text{b} \mid \text{b} \mid \text{b} \mid \text{b} \mid \text{b} \mid \text{b} \]

\[ \text{basic tests (} \uparrow \text{)} \]

ALPHABET \( \mathcal{A} = \mathbb{T} \cup \mathbb{P} \)

ASSERTION TELLS ME ABOUT THE STATE OF MY VARIABLES

VARIABLES \( t_1, \ldots, t_n, T_1, \ldots, T_n \)

\[ \alpha ::= t_1, t_2, t_3, \ldots, t_n, T_1, T_2, \ldots, T_n \] — FULL TEST / VALUATION

ONE FOR EACH VARIABLE

THESE ARE THE ATOMS OF \( \mathcal{B} \mathcal{A} \) GENERATED BY \( T \).
Suppose we have $t_1, t_2, \overline{t_1}, \overline{t_2}$

What information can I have?

$\alpha_0 = t_1, t_2$ each atom tells you exactly

$\alpha_1 = t_1, t_2$ what variable's we have.

$\alpha_2 = t_1, \overline{t_2}$ atoms $= 2^T \leq B$

$\alpha_3 = \overline{t_1}, t_2$

Semantics of Booleans $[b] = 2^T$

Regular Expressions ~ Traces

Trace for KAT programs:

$\alpha_0 P_0 \alpha_1 P_1 \alpha_2 P_2 \ldots \alpha_k P_k \overline{\alpha_{k+1}}$

Actions may change the state of variables

$A^* \mapsto (At \cdot P)^* \cdot At$

For example: $\overline{P} [x := 1]$ uninterpreted

$[p] = \alpha_1 \mapsto \alpha [x = 1]$

Pous (POPL'15): Symbolic program equivalence

- Tests as BDDs (Boolean decision diagrams)
- Programs as FODs (BDDs + variables)
GUARDED STRINGS

\[(\text{At } p)^* \cdot \text{At}\]

\[\llbracket e \rrbracket = \{ x \mid x \leq b \}\]

\[\llbracket b \rrbracket = \{ x \mid x \leq b \}\]

\[\llbracket p \rrbracket = \{ x \mid p \beta \mid x, \beta \in \text{At}\}\]

transform any \(\alpha\) to any \(\beta\)

DENOTATIONAL SEMANTICS

RELATIONAL SEMANTICS \((R(p) : 2^{\text{At} \times \text{At}})\) allows

FOR FURTHER REFINEMENT:

(BACK TO DENOTATIONAL SEMANTICS)

\[\llbracket e + f \rrbracket = \llbracket e \rrbracket \cup \llbracket f \rrbracket\]

\[\llbracket e ; f \rrbracket = \llbracket e \rrbracket \circ \llbracket f \rrbracket\]

\[\alpha_0 p_0 \alpha_1 p_1 ... \alpha_k p_k \alpha_k \beta_0 q_0 \beta_1 \beta_2 ...\]

= \[\alpha_0 p_0 ... \alpha_k q_0 \beta_1 \beta_2 ...\]

(if \(\alpha_k = \beta_0\))

DELETE \(\beta_0\) AND COMBINE IF \(\alpha_k = \beta_0\)

PARTIAL, UNDEFINED OTHERWISE

NOTE: \(\alpha \leq \beta \Rightarrow \alpha = \beta\) FOR \(\alpha, \beta \in \text{At}\)

\[\llbracket \text{if } b \text{ then } p \text{ else } q \rrbracket = \llbracket b ; p \rrbracket \cup \llbracket b ; q \rrbracket\]

\((b ; p + b ; q)\)

= \[\llbracket b \rrbracket \circ \llbracket p \rrbracket \cup \llbracket b \rrbracket \circ \llbracket q \rrbracket\]

= \[\{ x \mid x \leq b \} \circ \{ x \mid p \beta \mid x, \beta \in \text{At}\} ...\]

= \[\{ x \mid p \beta \mid x \leq b \} \cup \{ x \mid p \beta \mid x, \beta \in \text{At}\} ...\]

= \[\{ x \mid p \beta \mid x \leq b \} \cup \{ x \mid p \beta \mid x \leq b \} \]

= \[\{ x \mid p \beta \mid x \leq b \} \cup \{ x \mid p \beta \mid x \leq b \} \]
\[\begin{align*}
\text{while } b \text{ do } p \] \\
(b; p) = \frac{b}{\beta} \\
\leq b \leq b, \quad \alpha \leq b, \quad \beta \leq b
\end{align*}\]

\text{while true do skip, } \equiv 0

; p \leftarrow \text{can never see the effects}

\text{while true do } p \equiv 0

\uparrow \text{still can't observe effects of } p

\underline{Connection to Hoare Triples:}

\[1 \equiv b \leq c \equiv \epsilon \leq \epsilon \]

\text{Under precondition } b, \text{ if } c \text{ terminates, it}

\text{has post-condition } c,

\[\iff b \leq c \equiv \epsilon \]

\text{there are no traces of } b \leq c \text{ that don't}

\text{satisfy } c,

\[\equiv b \leq c \leq \epsilon \]
Homework Partial Solutions:

\[(x + y)^* \leq (x^*y)^* \]

\[
(x + y)^* \leq (x^*y)^* \]
\[\times \leq (x^*y)^* \times = x^* (y^*x)^* \]
\[
\forall x, 1 \leq x^* : 1 \leq x^*, 1 \leq (x^*y)^*, \]
\[1 \leq (x^*y)^* \times \]
\[
\times x^* (y^*x)^* \leq x^* (y^*x)^* \]
\[\times x^* \leq x^* \times \] (Unfold)
\[\times x^* \leq 1 + x^* \]
\[
(y^*x)^* (y^*x)^* \leq (y^*x)^* \leq x^* (y^*x)^* \]
\[1 + (x+y)^* x^* (y^*x)^* \leq x^* (y^*x)^* \]
\[
\exists \exists \]
\[
\exists \exists \]
Fixpoint
\[
\Rightarrow (x+y)^* \leq \exists = x^* (y^*x)^* = (x^*y)^* x^* \]

- Deciding with axioms is equivalent to deciding
- With DFAs, but DFAs is better for mechanization
- KAT gives useful intuition for working with programs
KAT

- Soundness + Completeness
  \( K A \times B A \)
- Automaton Model
  \[
  O \xrightarrow{\alpha, p} O
  \]
  \( \alpha_0, \alpha_1, \alpha_2, \ldots, \alpha_k \)

- Decision Procedure for Equivalence
  - BDDs (Pous) - Binary Decision Diagrams
  - Matrixes (Koren)
  - In PSPACE

\( \text{Union-Find Data Structure (Hopcroft, Tarjans)} \)

\( O(\log^c n) \)

Coinduction up-to (NFA, DFA, BZ, DER)

- Stop procedure earlier based on structure
  - Sound, weaker way of checking
  - Worst case, can't stop early, but can in many cases

(Bonchi/Pous, POPL '14)
Example:

```plaintext
while b do
  p;
  while c do
    q;
    (b p (c q) = \overline{z} (2) \overline{b})
  end
end

DISTRIBUTION, DENESTING, b \cdot p；((b+c)(c+2p)) \overline{b+c}

SUPING
```

NET KAT is a special KAT (official cat: CARBON, PPL’15/l6).
- Zoom in on sequential composition: CARBON EYES
- Previously, NET CORE
- Reasoning about networks

```
\[ \text{Fields} = \varepsilon, f_1, f_2, \ldots \varepsilon \]
\[ \text{Values} = v_0, \ldots, v_k \]

Nodes can test/set
- Values in packets
```

```plaintext
f_1, f_2, \ldots, f_k
```
Net KAT

\[ p \leftarrow n \]
\[ t \leftarrow n \]

Interpreted!

Example:

Packet filter:

- **SW** - switch
- **TYPE** - 
- **PT** - port
- **DST** - destination

Interpreted:

More equations hold:

\[ f \leftarrow n; f \leftarrow m = f \leftarrow m \]
\[ f = n; f \leftarrow n = f = n \]
\[ f \leftarrow n; f = n \equiv f \leftarrow n \]