CS 512, Spring 2018, Handout 03

Omega-Regular Expressions, Linear-Time Properties, Safety, Liveness, Invariance

Assaf Kfoury

25 January 2018

from *regular expressions* to *ω-regular expressions*

- regular expressions E over alphabet Σ can be specified by a BNF definition:

  $$E \ ::= \ \varnothing \mid \varepsilon \mid A \mid E_1 + E_2 \mid E_1 \cdot E_2 \mid E^* \qquad \text{where } A \in \Sigma$$

  E^+ is taken as an abbreviation of the regular expression $E \cdot E^*$.

- regular expressions E define regular languages $\mathscr{L}(E)$, by induction:

  $$\mathscr{L}(\varnothing) = \varnothing, \quad \mathscr{L}(\varepsilon) = \{\varepsilon\}, \quad \mathscr{L}(A) = \{A\},$$

  $$\mathscr{L}(E_1 + E_2) = \mathscr{L}(E_1) \cup \mathscr{L}(E_2),$$

  $$\mathscr{L}(E_1 \cdot E_2) = \mathscr{L}(E_1) \cdot \mathscr{L}(E_2),$$

  $$\mathscr{L}(E^*) = (\mathscr{L}(E))^*$$

from *regular expressions* to *ω-regular expressions*

- regular expressions E over alphabet Σ can be specified by a BNF definition:

  $$E \ ::= \ \varnothing \mid \varepsilon \mid A \mid E_1 + E_2 \mid E_1 \cdot E_2 \mid E^* \qquad \text{where } A \in \Sigma$$

  E^+ is taken as an abbreviation of the regular expression $E \cdot E^*$.

- regular expressions E define regular languages $\mathscr{L}(E)$, by induction:

  $$\mathscr{L}(\varnothing) = \varnothing, \quad \mathscr{L}(\varepsilon) = \{\varepsilon\}, \quad \mathscr{L}(A) = \{A\},$$

  $$\mathscr{L}(E_1 + E_2) = \mathscr{L}(E_1) \cup \mathscr{L}(E_2),$$

  $$\mathscr{L}(E_1 \cdot E_2) = \mathscr{L}(E_1) \cdot \mathscr{L}(E_2),$$

  $$\mathscr{L}(E^*) = (\mathscr{L}(E))^*$$

- every ω-regular expression G over alphabet Σ takes the form:

  $$G \ = \ E_1 \cdot (F_1)^\omega \ + \ \cdots \ + \ E_n \cdot (F_n)^\omega$$

  where E_i and F_i are regular expressions with $\varepsilon \notin \mathscr{L}(F_i)$.

- ω-regular expressions G define ω-regular languages $\mathscr{L}(G)$:

  $$\mathscr{L}(G) = \ \mathscr{L}(E_1) \cdot \mathscr{L}(F_1)^\omega \cup \cdots \cup \mathscr{L}(E_n) \cdot \mathscr{L}(F_n)^\omega$$

closure properties of ω-*regular expressions*

- for all regular languages L_1 and L_2 over the alphabet Σ, we have:

  $(L_1 \cup L_2)$ is a regular language (closure under set *union*)

  $(L_1 \cap L_2)$ is a regular language (closure under set *intersection*)

  $(\Sigma^* - L_1)$ is a regular language (closure under set *complementation*)

- for all ω-regular languages L_1 and L_2 over the alphabet Σ, we have:

  $(L_1 \cup L_2)$ is a ω-regular language (closure under set *union*)

  $(L_1 \cap L_2)$ is a ω-regular language (closure under set *intersection*)

  $(\Sigma^* - L_1)$ is a ω-regular language (closure under set *complementation*)

More details on regular and ω-regular languages are in the handout *Finite Automata and Büchi Automata*, click here to retrieve.

linear-time properties

- a trace over a set AP (of atomic propositions) is an ω-word/sequence in $(2^{\mathsf{AP}})^\omega$
- a linear-time property P specifies a set of admissible traces, *i.e.*, the traces that a transition system must exhibit or is *allowed to exhibit*
- *Traces*(TS) is the set of traces that a transition system TS *actually exhibits*
- transition system TS satisfies property P, denoted TS $\models P$, iff *Traces*(TS) $\subseteq P$

  "TS satisfies LT property P if all of TS's observable behaviors are admissible"

More details on the preceding defintions are in the handout
Properties of Transition Systems, click here to retrieve.

invariant properties

- linear-time property P over AP is an invariant if P has the form:

$$P = \left\{ A_0\, A_1\, A_2\, \cdots \in (2^{\mathsf{AP}})^\omega \;\middle|\; \text{forall } j \geqslant 0 \text{ it holds that } A_j \models \Phi \right\}$$

  where Φ is a propositional-logic formula over AP.

  Φ is called an invariant condition of P.

invariant properties

- linear-time property P over AP is an invariant if P has the form:

$$P = \left\{ A_0\, A_1\, A_2\, \cdots \in (2^{\mathsf{AP}})^\omega \;\middle|\; \text{forall } j \geqslant 0 \text{ it holds that } A_j \models \Phi \right\}$$

where Φ is a propositional-logic formula over AP.

Φ is called an invariant condition of P.

- **Fact**: If $\mathsf{TS} = (S, \mathsf{Act}, \rightarrow, I, \mathsf{AP}, L)$ is a transition system
and P is a linear-time property over AP with invariant condition Φ, then

$$\begin{aligned} \mathsf{TS} \models P \;\; &\text{iff} \;\; \text{for every path } \pi \text{ in TS, it holds that } \mathit{trace}(\pi) \in P \\ &\text{iff} \;\; \text{for every state } s \text{ in a path of TS, it holds that } L(s) \models \Phi \end{aligned}$$

"Φ is satisfied by every initial state and

by every state reachable from an initial state along an execution of TS"

safety properties

- a safety property may specify that an action/behavior/display can occur only after a prior condition is fulfilled.

  Example: at an automated teller machine (ATM), money can be withdrawn only after a correct PIN is entered.

  If the ATM allows money to be withdrawn without entering a correct PIN, we will say the ATM is *not safe* to use.

safety properties

- a safety property may specify that an action/behavior/display can occur only after a prior condition is fulfilled.

  Example: at an automated teller machine (ATM), money can be withdrawn only after a correct PIN is entered.

  If the ATM allows money to be withdrawn without entering a correct PIN, we will say the ATM is *not safe* to use.

- a safety property P specifies that, if a path violates P, then a finite prefix of P is already violating it.

  Example: money is withdrawn without entering a correct PIN before.

safety properties

- a safety property may specify that an action/behavior/display can occur only after a prior condition is fulfilled.

  Example: at an automated teller machine (ATM), money can be withdrawn only after a correct PIN is entered.

  If the ATM allows money to be withdrawn without entering a correct PIN, we will say the ATM is *not safe* to use.

- a safety property P specifies that, if a path violates P, then a finite prefix of P is already violating it.

  Example: money is withdrawn without entering a correct PIN before.

- a linear-time property P over AP is a safety property if for every "bad" trace $\sigma \in \left((2^{\mathsf{AP}})^{\omega} - P\right)$ there is a finite prefix σ' of σ such that:

$$P \cap \left\{ \sigma'' \in (2^{\mathsf{AP}})^{\omega} \;\middle|\; \sigma' \text{ is a prefix of } \sigma'' \right\} = \varnothing$$

Though written differently, an equivalent definition of safety property is in *Properties of Transition Systems* $\boxed{\text{click here}}$.

why liveness?

- safety properties specify that **"something bad never happens"**
- doing nothing easily fulfills a safety property, and will never lead to a "bad" situation

why liveness?

- safety properties specify that **"something bad never happens"**
- doing nothing easily fulfills a safety property, and will never lead to a "bad" situation

- safety properties are complemented by liveness properties, indicating that some progress is taking place
- liveness properties assert that **"something good will happen eventually"**

why liveness?

- safety properties specify that **"something bad never happens"**
- doing nothing easily fulfills a safety property, and will never lead to a "bad" situation

- safety properties are complemented by liveness properties, indicating that some progress is taking place
- liveness properties assert that **"something good will happen eventually"**

definition: for every $\sigma \in (2^{\mathsf{AP}})^\omega$, define

$$\mathsf{pref}(\sigma) \triangleq \left\{ \sigma' \in (2^{\mathsf{AP}})^* \;\middle|\; \sigma' \text{ is a finite prefix of } \sigma \right\}$$

- a linear-time property P over AP is a liveness property if $\mathsf{pref}(P) = (2^{\mathsf{AP}})^*$.

"a linear-time property P is a liveness property if

every finite word in $(2^{AP})^*$ can be extended to an infinite word in P."

(THIS PAGE INTENTIONALLY LEFT BLANK)