Queens' College Cambridge

Hoare Logic and Model Checking

Alistair O'Brien

Department of Computer Science

June 5, 2022

1 Hoare Logic

While

Syntax

Arith Terms	$E ::= N \mid X \mid E + E \mid E - E \mid E \times E$
Bool Terms	$B ::= \mathsf{true} \mid \mathsf{false} \mid B \wedge B \mid B \vee B \mid \neg B$ $\mid E = E \mid E \leq E \mid E \geq E$
Commands	$C ::= \mathsf{skip} \mid C_1; C_2 \mid X := E$ $\mid \mathsf{if}\ B\ \mathsf{then}\ C\ \mathsf{else}\ C \mid \mathsf{while}\ B\ \mathsf{do}\ C$
Variables	$\chi ::= X \mid x$
Terms	$t ::= \chi \mid f(t_1, \ldots, t_n)$ e.g. $+, -, \ldots$
Assertions	$P ::= \top \mid \bot \mid P \wedge P \mid P \vee P \mid P \rightarrow P$ $\mid t = t \mid p(t_1, \ldots, t_n)$ e.g. $\geq, \leq, \ldots$ $\mid \exists x.P \mid \forall x.P$
Stack	$s \in \mathsf{Var} \rightarrow \mathbb{Z}$

Operational Semantics

$\boxed{\mathcal{E}\left[\!\left[\cdot\right]\!\right](\cdot) : \mathsf{AExp} \times \mathsf{Stack} \rightarrow \mathbb{Z}}$

$$\begin{aligned}
\mathcal{E}\left[\!\left[N\right]\!\right](s) &= N \\
\mathcal{E}\left[\!\left[X\right]\!\right](s) &= s(X) \\
\mathcal{E}\left[\!\left[E_1 \odot E_2\right]\!\right](s) &= \mathcal{E}\left[\!\left[E_1\right]\!\right](s) \odot \mathcal{E}\left[\!\left[E_2\right]\!\right](s)
\end{aligned}$$

$\boxed{\mathcal{B}\left[\!\left[\cdot\right]\!\right](\cdot) : \mathsf{BExp} \times \mathsf{Stack} \rightarrow \mathbb{B}}$

$$\begin{aligned}
\mathcal{B}\left[\!\left[\mathsf{true}\right]\!\right](s) &= \top \\
\mathcal{B}\left[\!\left[\mathsf{false}\right]\!\right](s) &= \bot \\
\mathcal{B}\left[\!\left[B_1 \odot B_2\right]\!\right](s) &= \mathcal{B}\left[\!\left[B_1\right]\!\right](s) \odot \mathcal{B}\left[\!\left[B_2\right]\!\right](s) \\
\mathcal{B}\left[\!\left[E_1\ \mathcal{R}\ E_2\right]\!\right](s) &= \begin{cases} \top & \text{if } \mathcal{E}\left[\!\left[E_1\right]\!\right](s)\ \mathcal{R}\ \mathcal{E}\left[\!\left[E_2\right]\!\right](s) \\ \bot & \text{otherwise} \end{cases}
\end{aligned}$$

$\boxed{\langle C, s\rangle \rightsquigarrow \langle C, s\rangle}$

$$\frac{\mathcal{E}\left[\!\left[E\right]\!\right](s) = N}{\langle X := E, s\rangle \rightsquigarrow \langle \mathsf{skip}, s[X \mapsto N]\rangle}\ \textsc{Assign} \qquad \frac{\langle C_1, s\rangle \rightsquigarrow \langle C_1', s'\rangle}{\langle C_1; C_2, s\rangle \rightsquigarrow \langle C_1'; C_2, s'\rangle}\ \textsc{Seq}_1$$

$$\frac{}{\langle \mathsf{skip}; C, s\rangle \rightsquigarrow \langle C, s\rangle}\ \textsc{Seq}_2 \qquad \frac{\mathcal{B}\left[\!\left[B\right]\!\right](s) = \top}{\langle \mathsf{if}\ B\ \mathsf{then}\ C_1\ \mathsf{else}\ C_2, s\rangle \rightsquigarrow \langle C_1, s\rangle}\ \textsc{If}_1$$

$$\frac{\mathcal{B}\left[\!\left[B\right]\!\right](s) = \bot}{\langle \mathsf{if}\ B\ \mathsf{then}\ C_1\ \mathsf{else}\ C_2, s\rangle \rightsquigarrow \langle C_2, s\rangle}\ \textsc{If}_2 \qquad \frac{\mathcal{B}\left[\!\left[B\right]\!\right](s) = \bot}{\langle \mathsf{while}\ B\ \mathsf{do}\ C, s\rangle \rightsquigarrow \langle \mathsf{skip}, s\rangle}\ \textsc{While}_2$$

$$\frac{\mathcal{B}\left[\!\left[B\right]\!\right](s) = \top}{\langle \mathsf{while}\ B\ \mathsf{do}\ C, s\rangle \rightsquigarrow \langle C; \mathsf{while}\ B\ \mathsf{do}\ C, s\rangle}\ \textsc{While}_1$$

Semantics of Assertions and Hoare Triples

$\boxed{\left[\!\left[\cdot\right]\!\right](\cdot) : \mathsf{Term} \times \mathsf{Stack} \to \mathbb{Z}}$

$$\begin{aligned}
\left[\!\left[\chi\right]\!\right](s) &= s(\chi) \\
\left[\!\left[f(t_1, \ldots, t_n)\right]\!\right](s) &= \left[\!\left[f\right]\!\right](\left[\!\left[t_1\right]\!\right](s), \ldots, \left[\!\left[t_n\right]\!\right](s)) \\
\left[\!\left[E\right]\!\right](s) &= \mathcal{E}\left[\!\left[E\right]\!\right](s) && (E \text{ can occur inside } t)
\end{aligned}$$

$\boxed{\left[\!\left[\cdot\right]\!\right] : \mathsf{Assertion} \to \mathcal{P}(\mathsf{Stack})}$

$$\begin{aligned}
\left[\!\left[\top\right]\!\right] &= \mathsf{Stack} \\
\left[\!\left[\bot\right]\!\right] &= \emptyset \\
\left[\!\left[P \wedge Q\right]\!\right] &= \left[\!\left[P\right]\!\right] \cap \left[\!\left[Q\right]\!\right] \\
\left[\!\left[P \vee Q\right]\!\right] &= \left[\!\left[P\right]\!\right] \cup \left[\!\left[Q\right]\!\right] \\
\left[\!\left[P \to Q\right]\!\right] &= \{s \in \mathsf{Stack} : s \in \left[\!\left[P\right]\!\right] \implies s \in \left[\!\left[Q\right]\!\right]\} \\
\left[\!\left[t_1 = t_2\right]\!\right] &= \{s \in \mathsf{Stack} : \left[\!\left[t_1\right]\!\right](s) = \left[\!\left[t_2\right]\!\right](s)\} \\
\left[\!\left[p(t_1, \ldots, t_n)\right]\!\right] &= \{s \in \mathsf{Stack} : \left[\!\left[p\right]\!\right](\left[\!\left[t_1\right]\!\right](s), \ldots, \left[\!\left[t_n\right]\!\right](s))\} \\
\left[\!\left[\forall x. P\right]\!\right] &= \{s \in \mathsf{Stack} : \forall N \in \mathbb{Z}. s[x \mapsto N] \in \left[\!\left[P\right]\!\right]\} \\
\left[\!\left[\exists x. P\right]\!\right] &= \{s \in \mathsf{Stack} : \exists N \in \mathbb{Z}. s[x \mapsto N] \in \left[\!\left[P\right]\!\right]\}
\end{aligned}$$

$\boxed{\models \{P\}\ C\ \{Q\}}$

- Assuming command C is executed in initial state satisfying *precondition* P and C terminates, then the terminal state satisfies *postcondition* Q:

$$\models \{P\}\ C\ \{Q\} \iff \forall s, s' \in \mathsf{Stack}. s \in \left[\!\left[P\right]\!\right] \wedge \langle C, s\rangle \rightsquigarrow^* \langle \mathsf{skip}, s'\rangle \implies s' \in \left[\!\left[Q\right]\!\right]$$

$\boxed{\models [P]\ C\ [Q]}$

- Assuming command C is executed in initial state satisfying *precondition* P, then C terminates and any terminal state satisfies *postcondition* Q:

$$\begin{aligned}
\models [P]\ C\ [Q] \iff & \forall s \in \mathsf{Stack}. s \in \left[\!\left[P\right]\!\right] \\
& \implies \langle C, s\rangle \not\rightsquigarrow^\omega \wedge (\forall s' \in \mathsf{Stack}. \langle C, s\rangle \rightsquigarrow^* \langle \mathsf{skip}, s'\rangle \implies s' \in \left[\!\left[Q\right]\!\right])
\end{aligned}$$

Proof System

$\boxed{\vdash_{\text{FOL}} P}$ (See IB Logic and Proof)

$\boxed{\vdash \{P\}\ C\ \{Q\}}$

$$\frac{\vdash \{P \wedge B\}\ C_1\ \{Q\} \qquad \vdash \{P \wedge \neg B\}\ C_2\ \{Q\}}{\vdash \{P\}\ \textsf{if}\ B\ \textsf{then}\ C_1\ \textsf{else}\ C_2\ \{Q\}}\ \textsc{If} \qquad \frac{}{\vdash \{\{E/X\}P\}\ X := E\ \{P\}}\ \textsc{Assign}$$

$$\frac{}{\vdash \{P\}\ \textsf{skip}\ \{P\}}\ \textsc{Skip} \qquad \frac{\vdash \{P \wedge B\}\ C \{P\}}{\vdash \{P\}\ \textsf{while}\ B\ \textsf{do}\ C\ \{P \wedge \neg B\}}\ \textsc{While}$$

$$\frac{\vdash \{P\}\ C_1\ \{Q\} \qquad \vdash \{Q\}\ C_2\ \{R\}}{\vdash \{P\} C_1; C_2 \{Q\}}\ \textsc{Seq} \qquad \frac{\vdash \{P\}\ C\ \{Q\} \qquad \mathsf{mod}(C) \cap \mathsf{fv}(R) = \emptyset}{\vdash \{P \wedge R\}\ C\ \{Q \wedge R\}}\ \textsc{Constancy}$$

$$\frac{\vdash_{\text{FOL}} P_1 \to P_2 \qquad \vdash \{P_2\}\ C\ \{Q_2\} \qquad \vdash_{\text{FOL}} Q_2 \to Q_1}{\vdash \{P_1\}\ C\ \{Q_1\}}\ \textsc{Consequence}$$

$\boxed{\vdash [P]\ C\ [Q]}$

$$\frac{\vdash [P \wedge B \wedge t = n]\ C [P \wedge t < n] \qquad \vdash_{\text{FOL}} P \wedge B \to t \geq 0}{\vdash [P]\ \textsf{while}\ B\ \textsf{do}\ C\ [P \wedge \neg B]}\ \textsc{While}\ (n \in \textsc{Var})$$

Theorems

Lemma 1.0.1. (**Semantic Properties**) The following holds:

Termination Program steps to $\textsf{skip} \iff$ it doesn't diverge:

$$\forall C \in \mathsf{Cmd}, s \in \mathsf{Stack}. (\exists s' \in \mathsf{Stack}.\ \langle C, s\rangle \rightsquigarrow^* \langle \mathsf{skip}, s'\rangle) \iff \langle C, s\rangle \not\rightsquigarrow^\omega$$

where

$$\langle C, s\rangle \not\rightsquigarrow \iff \nexists c \in \mathsf{Config}.\ \langle C, s\rangle \rightsquigarrow c$$
$$\langle C, s\rangle \rightsquigarrow^\omega \iff \exists c \in \mathsf{Config}. C \rightsquigarrow^* c \wedge c \not\rightsquigarrow$$

Determinacy of While

$$\forall C, C', C'' \in \mathsf{Cmd}, s, s', s'' \in \mathsf{Stack}.\ \langle C, s\rangle \rightsquigarrow^* \langle C', s'\rangle \wedge \langle C, s\rangle \rightsquigarrow^* \langle C'', s''\rangle \implies \langle C', s'\rangle = \langle C'', s''\rangle$$

Proof. Termination follows from $c \not\rightsquigarrow \iff c = \langle \mathsf{skip}, s\rangle$. Proof of determinacy by *structural induction* on C.

Lemma 1.0.2. (**Substitution Lemmas**) The following holds:

(i) $\mathcal{E}\, [\![\{E_2/X\}E_1]\!]\, (s) = \mathcal{E}\, [\![E_1]\!]\, (s[X \mapsto \mathcal{E}\, [\![E_2]\!]\, (s)])$

(ii) $[\![\{E/X\}t]\!]\, (s) = [\![t]\!]\, (s[X \mapsto \mathcal{E}\, [\![E]\!]\, (s)])$

(iii) $s \in \llbracket \{E/X\}P \rrbracket \iff s[X \mapsto \mathcal{E}\llbracket E \rrbracket(s)] \in \llbracket P \rrbracket$

Proof. Proof by *structural induction* on E_1, t and P.

Theorem 1.0.1. (**Soundness**) If $\vdash \{P\}\ C\ \{Q\}$, then $\vDash \{P\}\ C\ \{Q\}$.

Proof. Proof by *rule induction* on $\vdash \{P\}\ C\ \{Q\}$.

- Hoare Logic is not complete due to incompleteness of FOL w/ arithmetic.

Theorem 1.0.2. (**Incompleteness**) Hoare Logic is *incomplete*:

$$\vDash \{P\}\ C\ \{Q\} \not\Longrightarrow \vdash \{P\}\ C\ \{Q\}.$$

Proof. Assume $\vDash \{P\}\ C\ \{Q\} \implies \vdash \{P\}\ C\{Q\}$. We wish to show the contradiction of $\vDash P \implies \vdash_{\text{FOL}} P$. Assume $\vDash P$, that is $\forall s.s \in \llbracket P \rrbracket$. So we have $\vDash \{\top\}\ \mathsf{skip}\ \{P\}$. By completeness we have $\vdash \{\top\}\ \mathsf{skip}\ \{P\}$, that is:

$$\frac{\overline{\vdash \{\top\}\ \mathsf{skip}\ \{\top\}} \qquad \vdash_{\text{FOL}} \top \to P}{\vdash \{\top\}\ \mathsf{skip}\ \{P\}}$$

So we have $\vdash_{\text{FOL}} \top \to P \iff \vdash_{\text{FOL}} P$. □

Definition 1.0.1. (**WLP and SP**) Weakest liberal precondition wlp and strong post-conditions sp s.t

$$\vdash_{\text{FOL}} P \to \mathsf{wlp}(C, Q) \iff \vdash \{P\}\ C\ \{Q\} \iff \vdash_{\text{FOL}} \mathsf{sp}(P, C) \to Q$$

- wlp and sp don't exist due to *while loops*:

$$\begin{aligned}
\mathsf{wlp}(\mathsf{skip}, Q) &= Q \\
\mathsf{wlp}(X := E, Q) &= \{E/X\}Q \\
\mathsf{wlp}(C_1; C_2, Q) &= \mathsf{wlp}(C_1, \mathsf{wlp}(C_2, Q)) \\
\mathsf{wlp}(\mathsf{if}\ B\ \mathsf{then}\ C_1\ \mathsf{else}\ C_2, Q) &= (B \to \mathsf{wlp}(C_1, Q)) \land (\neg B \to \mathsf{wlp}(C_2, Q)) \\
\mathsf{wlp}(\mathsf{while}\ B\ \mathsf{do}\ C, Q) &= \mathsf{wlp}(\mathsf{if}\ B\ \mathsf{then}\ (C;\ \mathsf{while}\ B\ \mathsf{do}\ C)\ \mathsf{else}\ \mathsf{skip}, Q) \\
&= (B \to \mathsf{wlp}(C, \mathsf{wlp}(\mathsf{while}\ B\ \mathsf{do}\ C, Q))) \land (\neg B \to Q)
\end{aligned}$$

Theorem 1.0.3. (**Relative Completeness**) Hoare Logic is *relatively complete*:

$$\vDash \{P\}\ C\ \{Q\} \implies (\vdash \{P\}\ C\ \{Q\} \iff \text{completeness of } \vdash_{\text{FOL}}),$$

that is to say failure to derive $\vdash \{P\}\ C\ \{Q\}$ is only due to failure to derive $\vdash_{\text{FOL}} R$ (for valid R).

Theorem 1.0.4. (**Undecidability**) Hoare Logic is *undecidable*:

$$\nexists f \in \mathsf{Computable}. f(P, C, Q) = \top \iff \vDash \{P\}\ C\ \{Q\}.$$

Proof. Proof by *reduction to Halting problem*, by using $\vDash \{\top\}\ C\ \{\bot\}$ as a termination checker for C.

$\textsf{While}_p$

Syntax

Arith Terms	$E ::= N \mid X \mid E + E \mid E - E \mid E \times E$
Bool Terms	$B ::= \mathsf{true} \mid \mathsf{false} \mid B \land B \mid B \lor B \mid \neg B$ $\mid E = E \mid E \leq E \mid E \geq E$
Null	$\mathsf{null} = 0$
Commands	$C ::= \mathsf{skip} \mid C_1; C_2 \mid X := E \mid X := [E] \mid [E_1] := E_2$ $\mid \mathsf{if}\ B\ \mathsf{then}\ C\ \mathsf{else}\ C \mid \mathsf{while}\ B\ \mathsf{do}\ C$ $\mid X := \mathsf{alloc}(E_0, \dots, E_n) \mid \mathsf{dispose}(E)$
Variables	$\chi ::= X \mid x$
Terms	$t ::= \chi \mid f(t_1, \dots, t_n)$ e.g. $+, -, \dots$
Assertions	$P ::= \top \mid \bot \mid P \land P \mid P \lor P \mid P \to P$ $\mid t = t \mid p(t_1, \dots, t_n)$ e.g. $\geq, \leq, \dots$ $\mid \exists x.P \mid \forall x.P$ $\mid t \mapsto t \mid P * P \mid \mathsf{emp}$
Stack	$s \in \mathsf{Var} \to \mathbb{Z}$
Location	$\ell \in \mathsf{Loc} = \mathbb{Z}_{\geq 0}$
Heap	$h \in \mathsf{Heap} = (\mathsf{Loc} \setminus \{\mathsf{null}\}) \rightharpoonup \mathbb{Z}$
State	$\sigma ::= \langle s, h \rangle$
Config	$c ::= \langle C, \sigma \rangle \mid \lightning$

Operational Semantics

$\boxed{\mathcal{E}[\![\cdot]\!](\cdot) : \mathsf{AExp} \times \mathsf{Stack} \to \mathbb{Z}}$

$$\begin{aligned} \mathcal{E}[\![N]\!](s) &= N \\ \mathcal{E}[\![X]\!](s) &= s(X) \\ \mathcal{E}[\![E_1 \odot E_2]\!](s) &= \mathcal{E}[\![E_1]\!](s) \odot \mathcal{E}[\![E_2]\!](s) \end{aligned}$$

$\boxed{\mathcal{B}[\![\cdot]\!](\cdot) : \mathsf{BExp} \times \mathsf{Stack} \to \mathbb{B}}$

$$\begin{aligned} \mathcal{B}[\![\mathsf{true}]\!](s) &= \top \\ \mathcal{B}[\![\mathsf{false}]\!](s) &= \bot \\ \mathcal{B}[\![B_1 \odot B_2]\!](s) &= \mathcal{B}[\![B_1]\!](s) \odot \mathcal{B}[\![B_2]\!](s) \\ \mathcal{B}[\![E_1\ \mathcal{R}\ E_2]\!](s) &= \begin{cases} \top & \text{if } \mathcal{E}[\![E_1]\!](s)\ \mathcal{R}\ \mathcal{E}[\![E_2]\!](s) \\ \bot & \text{otherwise} \end{cases} \end{aligned}$$

$\boxed{c \rightsquigarrow c}$

$$\frac{\mathcal{E}\,[\![E]\!]\,(s) = N}{\langle X := E, \langle s, h\rangle\rangle \rightsquigarrow \langle \mathsf{skip}, \langle s[X \mapsto N], h\rangle\rangle}\ \textsc{Assign} \qquad \frac{\langle C_1, \sigma\rangle \rightsquigarrow \langle C_1', \sigma'\rangle}{\langle C_1; C_2, \sigma\rangle \rightsquigarrow \langle C_1'; C_2, \sigma'\rangle}\ \textsc{Seq}_1$$

$$\frac{}{\langle \mathsf{skip}; C, \sigma\rangle \rightsquigarrow \langle C, \sigma\rangle}\ \textsc{Seq}_2 \qquad \frac{\mathcal{B}\,[\![B]\!]\,(s) = \top}{\langle \mathsf{if}\ B\ \mathsf{then}\ C_1\ \mathsf{else}\ C_2, \langle s, h\rangle\rangle \rightsquigarrow \langle C_1, \langle s, h\rangle\rangle}\ \textsc{If}_1$$

$$\frac{\mathcal{B}\,[\![B]\!]\,(s) = \bot}{\langle \mathsf{if}\ B\ \mathsf{then}\ C_1\ \mathsf{else}\ C_2, \langle s, h\rangle\rangle \rightsquigarrow \langle C_2, \langle s, h\rangle\rangle}\ \textsc{If}_2 \qquad \frac{\mathcal{B}\,[\![B]\!]\,(s) = \bot}{\langle \mathsf{while}\ B\ \mathsf{do}\ C, \langle s, h\rangle\rangle \rightsquigarrow \langle \mathsf{skip}, \langle s, h\rangle\rangle}\ \textsc{While}_2$$

$$\frac{\mathcal{B}\,[\![B]\!]\,(s) = \top}{\langle \mathsf{while}\ B\ \mathsf{do}\ C, \langle s, h\rangle\rangle \rightsquigarrow \langle C; \mathsf{while}\ B\ \mathsf{do}\ C, \langle s, h\rangle\rangle}\ \textsc{While}_1 \qquad \frac{\mathcal{E}\,[\![E]\!]\,(s) = \ell \in \mathsf{Loc} \qquad \ell \in \operatorname{dom} h}{\langle s, h\rangle \vdash_{\textsc{Loc}} E}\ \textsc{Loc}$$

$$\frac{\mathcal{E}\,[\![E]\!]\,(s) = \ell \qquad \ell \in \operatorname{dom} h \qquad h(\ell) = N}{\langle X := [E], \langle s, h\rangle\rangle \rightsquigarrow \langle \mathsf{skip}, \langle s[X \mapsto N], h\rangle\rangle}\ \textsc{Deref}_1 \qquad \frac{\sigma \not\vdash_{\textsc{Loc}} E}{\langle X := [E], \sigma\rangle \rightsquigarrow \lightning}\ \textsc{Deref}_2$$

$$\frac{\mathcal{E}\,[\![E_1]\!]\,(s) = \ell \qquad \ell \in \operatorname{dom} h \qquad \mathcal{E}\,[\![E_2]\!]\,(s) = N}{\langle [E_1] := E_2, \langle s, h\rangle\rangle \rightsquigarrow \langle \mathsf{skip}, \langle s, h[\ell \mapsto N]\rangle\rangle}\ \textsc{HAssign}_1 \qquad \frac{\sigma \not\vdash_{\textsc{Loc}} E_1}{\langle [E_1] := E_2, \sigma\rangle \rightsquigarrow \lightning}\ \textsc{HAssign}_2$$

$$\frac{\mathcal{E}\,[\![E]\!]\,(s) = \ell \qquad \ell \in \operatorname{dom} h}{\langle \mathsf{dispose}(E), \langle s, h\rangle\rangle \rightsquigarrow \langle \mathsf{skip}, \langle s, h \setminus \ell\rangle\rangle}\ \textsc{Dispose}_1 \qquad \frac{\sigma \not\vdash_{\textsc{Loc}} E}{\langle \mathsf{dispose}(E), \sigma\rangle \rightsquigarrow \lightning}\ \textsc{Dispose}_2$$

$$\frac{\mathcal{E}\,[\![E_i]\!]\,(s) = N_i \qquad \forall i \in [0, n].\ell + i \notin \operatorname{dom} h \qquad \ell \neq \mathsf{null}}{\langle X := \mathsf{alloc}(E_0, \dots, E_n), \langle s, h\rangle\rangle \rightsquigarrow \langle \mathsf{skip}, \langle s[X \mapsto \ell], h[\ell \mapsto N_0, \dots, \ell + n \mapsto N_n]\rangle\rangle}\ \textsc{Alloc}$$

Semantics of Assertions and Hoare Triples

$\boxed{[\![\cdot]\!]\,(\cdot) : \mathsf{Term} \times \mathsf{Stack} \to \mathbb{Z}}$

$$\begin{aligned} [\![\chi]\!]\,(s) &= s(\chi) \\ [\![f(t_1, \dots, t_n)]\!]\,(s) &= [\![f]\!]\,([\![t_1]\!]\,(s), \dots, [\![t_n]\!]\,(s)) \\ [\![E]\!]\,(s) &= \mathcal{E}\,[\![E]\!]\,(s) && (E \text{ can occur inside } t) \end{aligned}$$

$\boxed{[\![\cdot]\!]\,(\cdot) : \mathsf{Assertion} \times \mathsf{Stack} \to \mathcal{P}(\mathsf{Heap})}$

$$\begin{aligned} [\![\top]\!]\,(s) &= \mathsf{Heap} \\ [\![\bot]\!]\,(s) &= \emptyset \\ [\![P \wedge Q]\!]\,(s) &= [\![P]\!]\,(s) \cap [\![Q]\!]\,(s) \\ [\![P \vee Q]\!]\,(s) &= [\![P]\!]\,(s) \cup [\![Q]\!]\,(s) \\ [\![P \to Q]\!]\,(s) &= \{h \in \mathsf{Heap} : h \in [\![P]\!]\,(s) \implies h \in [\![Q]\!]\,(s)\} \\ [\![t_1 = t_2]\!]\,(s) &= \begin{cases} \mathsf{Heap} & \text{if } [\![t_1]\!]\,(s) = [\![t_2]\!]\,(s) \\ \emptyset & \text{otherwise} \end{cases} \\ [\![p(t_1, \dots, t_n)]\!]\,(s) &= \begin{cases} \mathsf{Heap} & \text{if } [\![p]\!]\,([\![t_1]\!]\,(s), \dots, [\![t_n]\!]\,(s)) \\ \emptyset & \text{otherwise} \end{cases} \end{aligned}$$

$$
\begin{aligned}
[\![\forall x.P]\!]\,(s) &= \{h \in \mathsf{Heap} : \forall N \in \mathbb{Z}.h \in [\![P]\!]\,(s[x \mapsto N])\} \\
[\![\exists x.P]\!]\,(s) &= \{h \in \mathsf{Heap} : \exists N \in \mathbb{Z}.h \in [\![P]\!]\,(s[x \mapsto N])\} \\
[\![t_1 \mapsto t_2]\!]\,(s) &= \left\{ h \in \mathsf{Heap} : \exists \ell \in \mathsf{Loc}, N \in \mathbb{Z}. \begin{array}{l} [\![t_1]\!]\,(s) = \ell \wedge [\![t_2]\!]\,(s) = N \wedge \ell \neq \mathsf{null} \\ \wedge \operatorname{dom} h = \{\ell\} \wedge h(\ell) = N \end{array} \right\} \\
[\![P * Q]\!]\,(s) &= \{h \in \mathsf{Heap} : \exists h_1, h_2 \in \mathsf{Heap}.h_1 \in [\![P]\!]\,(s) \wedge h_2 \in [\![Q]\!]\,(s) \wedge h = h_1 \uplus h_2\} \\
[\![\mathsf{emp}]\!]\,(s) &= \{h \in \mathsf{Heap} : \operatorname{dom} h = \emptyset\}
\end{aligned}
$$

- $t_1 \mapsto t_2$ asserts ownership of t_1 (ℓ) with value t_2 in heap ($\operatorname{dom} h = \{\ell\}$).
- $P * Q$ splits heap into disjoint parts h_1, h_2 satisfying P and Q resp.
- emp asserts ownership of no locations (empty heap).

$\boxed{\models \{P\}\ C\ \{Q\}}$

- Assuming state $\langle s, h_1 \rangle$ satisfies *precondition* P and command C executed in initial state $\langle s, h_1 \uplus h_F \rangle$ (h_F = unchanged frame heaplet), then:
  1. C does not fault (P asserts ownership of all locations accessed),
  2. if C terminates then terminal state is $\langle s', h_1' \uplus h_F \rangle$ and $\langle s', h_1' \rangle$ satisfies *postcondition* Q.

$$
\begin{aligned}
\models \{P\}\ C\ \{Q\} \iff\ & \forall s \in \mathsf{Stack}, h_1, h_F \in \mathsf{Heap}. \operatorname{dom} h_1 \cap \operatorname{dom} h_F = \emptyset \wedge h_1 \in [\![P]\!]\,(s) \\
& \implies \langle C, \langle s, h_1 \uplus h_F \rangle \rangle \not\rightsquigarrow^* \lightning \\
& \quad \wedge \forall s' \in \mathsf{Stack}, h' \in \mathsf{Heap}. \langle C, \langle s, h_1 \uplus h_F \rangle \rangle \rightsquigarrow^* \langle \mathsf{skip}, \langle s', h' \rangle \rangle \\
& \qquad \implies \exists h_1' \in \mathsf{Heap}.h' = h_1' \uplus h_F \wedge h_1' \in [\![Q]\!]\,(s)
\end{aligned}
$$

Proof System

$\boxed{\vdash_{\mathrm{CSL}} P}$ (Separation Logic Proof System)

$\boxed{\vdash \{P\}\ C\ \{Q\}}$

$$
\frac{\vdash \{P \wedge B\}\ C_1\ \{Q\} \qquad \vdash \{P \wedge \neg B\}\ C_2\ \{Q\}}{\vdash \{P\}\ \mathsf{if}\ B\ \mathsf{then}\ C_1\ \mathsf{else}\ C_2\ \{Q\}}\ \textsc{If}
\qquad
\frac{}{\vdash \{\{E/X\}P\}\ X := E\ \{P\}}\ \textsc{Assign}
$$

$$
\frac{}{\vdash \{P\}\ \mathsf{skip}\ \{P\}}\ \textsc{Skip}
\qquad
\frac{\vdash \{P \wedge B\}\ C\{P\}}{\vdash \{P\}\ \mathsf{while}\ B\ \mathsf{do}\ C\ \{P \wedge \neg B\}}\ \textsc{While}
$$

$$
\frac{\vdash \{P\}\ C_1\ \{Q\} \qquad \vdash \{Q\}\ C_2\ \{R\}}{\vdash \{P\}C_1; C_2\{Q\}}\ \textsc{Seq}
\qquad
\frac{\vdash \{P\}\ C\ \{Q\} \qquad \mathsf{mod}(C) \cap \mathsf{fv}(R) = \emptyset}{\vdash \{P \wedge R\}\ C\ \{Q \wedge R\}}\ \textsc{Constancy}
$$

$$
\frac{\vdash_{\mathrm{CSL}} P_1 \to P_2 \qquad \vdash \{P_2\}\ C\ \{Q_2\} \qquad \vdash_{\mathrm{CSL}} Q_2 \to Q_1}{\vdash \{P_1\}\ C\ \{Q_1\}}\ \textsc{Consequence}
$$

$$\frac{\vdash \{P\}\ C\ \{Q\} \qquad \mathsf{mod}(C) \cap \mathsf{fv}(R) = \emptyset}{\vdash \{P * R\}\ C\ \{Q * R\}}\ \textsc{Framing} \qquad \frac{\vdash \{P\}\ C\ \{Q\}}{\vdash \{\exists x.P\}\ C\ \{\exists x.Q\}}\ \textsc{Exists}$$

$$\frac{}{\vdash \{E \mapsto t\}\ \mathsf{dispose}(E)\ \{\mathsf{emp}\}}\ \textsc{Dispose} \qquad \frac{}{\vdash \{E_1 \mapsto t\}\ [E_1] := E_2\ \{E_1 \mapsto E_2\}}\ \textsc{HAssign}$$

$$\frac{}{\vdash \{E \mapsto v \wedge X = x\}\ X := [E]\ \{\{x/X\}E \mapsto v \wedge X = v\}}\ \textsc{Deref}$$

$$\frac{}{\vdash \{X = x \wedge \mathsf{emp}\}\ X := \mathsf{alloc}(E_0, \ldots, E_n)\ \{X \mapsto \{x/X\}E_0, \ldots, \{x/X\}E_n\}}\ \textsc{Alloc}$$

2 Model Checking

Temporal Models

Definition 2.0.1. (**Temporal Model**) A temporal model $M \in \mathsf{TModel}(\mathsf{AP})$ over atomic propositions AP is a tuple $(S, S_0, \cdot \to \cdot, \ell)$ where:

- S is a set of states and $S_0 \subseteq S$ is the initial set of states,
- $\cdot \to \cdot : S \nrightarrow S$ is the *accessibility/transition relation*
- $\ell : S \to \mathcal{P}(\mathsf{AP})$ is a labelling function

and $\cdot \to \cdot$ is *left-total*:

$$\forall s \in S. \exists s' \in S. s \to s'$$

Definition 2.0.2. (**Path**) A path π through M is mapping $\pi : \mathbb{N} \to S$ satisfying:

$$\forall n \in \mathbb{N}. \pi(n) \to \pi(n+1)$$

$\pi \setminus n$ is *suffix* of path π, defined by $i \mapsto \pi(i+n)$

Definition 2.0.3. (**Reachable**) The set of reachable states in M is defined as

$$\mathsf{Reachable}(M) = \{\pi \in \mathsf{Path}(M) : \pi(0) \in S_0\}$$

$s \in S$ is reachable in M if $s \in \mathsf{Reachable}(M)$, that is:

$$\exists \pi \in \mathsf{Path}(M), n \in \mathbb{N}. \pi(0) \in S_0 \land \pi(n) = s$$

Definition 2.0.4. (**Stuttering**) A model M is said to be *stuttering* iff

$$\forall s \in S. s \to s$$

- **Definite Temporal Models**:
  - S and S_0 are finite
  - $\cdot \to \cdot$ and ℓ are computable

Temporal Logics

(Linear Temporal Logic) LTL

- Describes properties of **paths** in models
- Considers infinite **linear** paths (each state has exactly 1 *successor*)
- Cannot reason about *states* (or their successors/branching)

Syntax

Path Property	$\phi ::= \bot$	(False: no path satisfies)
	$\mid \top$	(True: all paths satisfy)
	$\mid p$	(Atomic predicate: head satisfies p)
	$\mid \neg\phi$	(Negation: path doesn't satisfy ϕ)
	$\mid \phi_1 \wedge \phi_2$	(Conjunction: path satisfies ϕ_1 and ϕ_2)
	$\mid \phi_1 \vee \phi_2$	(Disjunction: path satisfies ϕ_1 or ϕ_2)
	$\mid \phi_1 \to \phi_2$	(Implication: path satisfies ϕ_1 then satisfies ϕ_2)
	$\mid \mathsf{X}\ \phi$	(neXt: tail satisfies ϕ)
	$\mid \mathsf{G}\ \phi$	(Generally: every path suffix satisfies ϕ)
	$\mid \mathsf{F}\ \phi$	(Future: some path suffix satisfies ϕ)
	$\mid \phi_1\ \mathsf{U}\ \phi$	(Until: some path suffix satisfies ϕ_2, all prefixes until then satisfy ϕ_1)

Semantics

$\boxed{M \models \phi}$, $\boxed{s \models_M \phi}$

$$
\begin{aligned}
M \models \phi &= \forall s \in S. s \in S_0 \implies s \models_M \phi \\
s \models_M \phi &= \forall \pi \in \mathsf{Path}(M). \pi(0) = s \implies \pi \models_M \phi
\end{aligned}
$$

$\boxed{\pi \models_M \phi}$

$$
\begin{aligned}
\pi \models_M \top &= \top \\
\pi \models_M \top &= \bot \\
\pi \models_M p &= p \in \ell(\pi(0)) \\
\pi \models_M \neg\phi &= \neg(\pi \models_M \phi) \\
\pi \models_M \phi_1 \wedge \phi_2 &= \pi \models_M \phi_1 \wedge \phi \models_M \phi_2 \\
\pi \models_M \phi_1 \vee \phi_2 &= \pi \models_M \phi_1 \vee \phi \models_M \phi_2 \\
\pi \models_M \phi_1 \to \phi_2 &= \neg(\pi \models_M \phi_1) \vee \pi \models_M \phi_2 \\
\pi \models_M \mathsf{X}\ \phi &= \pi \setminus 1 \models_M \phi \\
\pi \models_M \mathsf{F}\ \phi &= \exists n \in \mathbb{N}. \pi \setminus n \models_M \phi \\
\pi \models_M \mathsf{G}\ \phi &= \forall n \in \mathbb{N}. \pi \setminus n \models_M \phi \\
\pi \models_M \phi_1\ \mathsf{U}\ \phi_2 &= \exists n \in \mathbb{N}. \pi \setminus n \models_M \phi_2 \wedge (\forall k \in [0, n). \pi \setminus k \models_M \phi_1)
\end{aligned}
$$

(Computation Tree Logic) CTL*

- Describes properties of possible *path trees*
- Considers *set* of possible futures for each state

Syntax

State Property
$$
\begin{aligned}
\psi ::=\ & \bot && \text{(False: no state satisfies)}\\
\mid\ & \top && \text{(True: all states satisfy)}\\
\mid\ & p && \text{(Atomic predicate: state satisfies } p\text{)}\\
\mid\ & \psi_1 \wedge^s \psi_2 && \text{(Conjunction: state satisfies } \psi_1 \text{ and } \psi_2\text{)}\\
\mid\ & \psi_1 \vee^s \psi_2 && \text{(Disjunction: state satisfies } \psi_1 \text{ or } \psi_2\text{)}\\
\mid\ & \psi_1 \rightarrow^s \psi_2 && \text{(Implication: state satisfies } \psi_1 \text{ then satisfies } \psi_2\text{)}\\
\mid\ & \mathsf{A}\ \phi && \text{(Universal: every outgoing path satisfies } \phi\text{)}\\
\mid\ & \mathsf{E}\ \phi && \text{(Existential: some outgoing path satisfies } \phi\text{)}
\end{aligned}
$$

Path Property
$$
\begin{aligned}
\phi ::=\ & \psi && \text{(State: head satisfies } \psi\text{)}\\
\mid\ & \phi_1 \wedge \phi_2 && \text{(Conjunction: path satisfies } \phi_1 \text{ and } \phi_2\text{)}\\
\mid\ & \phi_1 \vee \phi_2 && \text{(Disjunction: path satisfies } \phi_1 \text{ or } \phi_2\text{)}\\
\mid\ & \phi_1 \rightarrow \phi_2 && \text{(Implication: path satisfies } \phi_1 \text{ then satisfies } \phi_2\text{)}\\
\mid\ & \mathsf{X}\ \phi && \text{(neXt: tail satisfies } \phi\text{)}\\
\mid\ & \mathsf{G}\ \phi && \text{(Generally: every path suffix satisfies } \phi\text{)}\\
\mid\ & \mathsf{F}\ \phi && \text{(Future: some path suffix satisfies } \phi\text{)}\\
\mid\ & \phi_1\ \mathsf{U}\ \phi_2 && \text{(Until: some path suffix satisfies } \phi_2\text{,}\\
& && \text{all prefixes until then satisfy } \phi_1\text{)}
\end{aligned}
$$

Semantics

$\boxed{M \vDash \psi}$

$$M \vDash \psi = \forall s \in S. s \in S_0 \implies s \vDash_M \psi$$

$\boxed{s \vDash_M \psi}$, $\boxed{\pi \vDash_M \phi}$

$$
\begin{aligned}
s \vDash_M \top &= \top\\
s \vDash_M \bot &= \bot\\
s \vDash_M p &= p \in \ell(s)\\
s \vDash_M \psi_1 \wedge^s \psi_2 &= s \vDash_M \psi_1 \wedge s \vDash_M \psi_2\\
s \vDash_M \psi_1 \vee^s \psi_2 &= s \vDash_M \psi_1 \vee s \vDash_M \psi_2\\
s \vDash_M \psi_1 \rightarrow^s \psi_2 &= \neg(s \vDash_M \psi_1) \vee s \vDash_M \psi_2\\
s \vDash_M \mathsf{A}\ \phi &= \forall \pi \in \mathsf{Path}(M). \pi(0) = s \implies \pi \vDash_M \phi\\
s \vDash_M \mathsf{E}\ \phi &= \exists \pi \in \mathsf{Path}(M). \pi(0) = s \implies \pi \vDash_M \phi
\end{aligned}
$$

$$
\begin{aligned}
\pi \vDash_M \psi &= \pi(0) \vDash_M \psi\\
\pi \vDash_M \phi_1 \wedge^p \phi_2 &= \pi \vDash_M \phi_1 \wedge \pi \vDash_M \phi_2\\
\pi \vDash_M \phi_1 \vee^p \phi_2 &= \pi \vDash_M \phi_1 \vee \pi \vDash_M \phi_2\\
\pi \vDash_M \phi_1 \rightarrow \phi_2 &= \neg(\pi \vDash_M \phi_1) \vee \pi \vDash_M \phi_2\\
\pi \vDash_M \mathsf{X}\ \phi &= \pi \setminus 1 \vDash_M \phi\\
\pi \vDash_M \mathsf{F}\ \phi &= \exists n \in \mathbb{N}. \pi \setminus n \vDash \phi\\
\pi \vDash_M \mathsf{G}\ \phi &= \forall n \in \mathbb{N}. \pi \setminus n \vDash_M \phi\\
\pi \vDash_M \phi_1\ \mathsf{U}\ \phi_2 &= \exists n \in \mathbb{N}. \pi \setminus n \vDash_M \phi_2\\
&\quad \wedge (\forall k \in [0, n). \pi \setminus k \vDash_M \phi_1)
\end{aligned}
$$

- CLT* fragments:

CTL Force all temporal operators $(\mathsf{X}, \mathsf{F}, \mathsf{G}, \mathsf{U})$ to use path quantifiers (A, E)

LTL No path quantifiers, no explicit state props, uses A implicitly.

ACTL^* Universal fragment of CTL^*

ECTL^* Existential fragment of CTL^*

Automated Theorem Proving

Simulations

- Concrete temporal model transformed to abstract model with *reduced* state space

Definition 2.0.5. (**Simulation**) Let $M = (S, S_0, \cdot \to \cdot, \ell) \in \mathsf{TModel}(\mathsf{AP})$ and $M' = (S', S'_0, \cdot \rightsquigarrow \cdot, \ell') \in \mathsf{TModel}(\mathsf{AP}')$ be temporal models where $\mathsf{AP}' \subseteq \mathsf{AP}$. $R \subseteq S \times S'$ is a *simulation*, denoted $M \preceq^R M'$ if:

(i) R is consistent w/ labels:

$$\forall s \in S, s' \in S'.s \; R \; s' \implies \ell(s) \cap \mathsf{AP}' = \ell'(s')$$

(ii) R is consistent w/ initial states:

$$\forall s \in S_0.\exists s' \in S'_0.s \; R \; s'$$

(iii) Any step M has a corresponding step in M' for any R-related start and end states:

$$\forall s_1, s_2 \in S, s'_1 \in S'.s_1 \; R \; s'_1 \wedge s_1 \to s_2 \implies \exists s'_2 \in S'.s'_1 \rightsquigarrow s'_2 \wedge s_2 \; R \; s'_2$$

Definition 2.0.6. (**Simulation Preorder**) The simulation preorder $\cdot \preceq \cdot$ is defined as:

$$M \preceq M' = \exists R \subseteq S \times S'.M \preceq^R M'$$

Theorem 2.0.1. (**Simulation preserves ACTL^***) The universal, implication-free fragment of CTL^* ($\mathsf{ACTL}^{*\mathsf{IF}}$) is consistent w/ simulation:

$$\begin{aligned}\forall M \in& \mathsf{TModel}(\mathsf{AP}), M' \in \mathsf{TModel}(\mathsf{AP}'), \psi \in \mathsf{StateProp}^{\mathsf{ACTL}^{*\mathsf{IF}}}, \\ &M \preceq M' \wedge M' \vDash \psi \implies M \vDash \psi\end{aligned}$$

- **Problem**: $M \nvDash \psi \;\not\Longrightarrow\; M \nvDash \psi$.

Definition 2.0.7. (**Bisimulation**) Let $M = (S, S_0, \cdot \to \cdot, \ell) \in \mathsf{TModel}(\mathsf{AP})$ and $M' = (S', S'_0, \cdot \rightsquigarrow \cdot, \ell') \in \mathsf{TModel}(\mathsf{AP})$. $R \subseteq S \times S'$ is a *bisimulation*, denoted $M \approx^R M'$ if:

(i) R is consistent w/ labels:

$$\forall s \in S, s' \in S'.s \; R \; s' \implies \ell(s) = \ell'(s')$$

(ii) R bi-directionally relates initial states:

$$(\forall s \in S_0.\exists s' \in S'_0.s \; R \; s') \wedge (\forall s' \in S'_0.\exists s \in S_0.s \; R \; s')$$

(iii) • M can match steps of M':

$$\forall s_1, s_2 \in S, s_1' \in S'.s_1 \ R \ s_1' \wedge s_1 \to s_2 \implies \exists s_2' \in S'.s_1' \rightsquigarrow s_2' \wedge s_2 \ R \ s_2'$$

• M' can match steps of M:

$$\forall s_1', s_2' \in S', s_1 \in S.s_1 \ R \ s_1' \wedge s_1' \rightsquigarrow s_2' \implies \exists s_2 \in S.s_1 \to s_2 \wedge s_2 \ R \ s_2'$$

Definition 2.0.8. (**Bisimilarity**) M and M' are *bisimilar*, denoted $M \approx M'$, give by:

$$M \approx M' = \exists R \subseteq S \times S'.M \approx^R M'$$

We have

$$M \approx M' \implies M \preceq M' \wedge M' \preceq M$$

Theorem 2.0.2. (**Bisimulation preserves CTL^***) CTL^* is consistent w/ bisimulation:

$$\begin{aligned}&\forall M, M' \in \mathsf{TModel}(\mathsf{AP}), \psi \in \mathsf{StateProp}.\\ &\quad M \approx M' \implies (M' \vDash \psi \iff M \vDash \psi)\end{aligned}$$

Model Checker

- **Idea**: Function `mca` that computes set of states that a state prop satisfies. Focusing on ECTL (using dual laws for CTL).
- **Problems**: Approach is slow.
- **Solutions**: Memoization, "symbolic model checking" (BBDs, etc), lazy computations, partial orderings (reduces # repeated computations), etc

```
open Core
(* States are integers n ≥ 0 *)
module State = Int

(* Temporal model parameterized by atomic props ['ap].
   Assumed to be left total *)
type 'ap tmodel =
  { s : State.Set.t
  ; s0 : state -> bool
  ; t : state -> state -> bool
  ; l : state -> 'ap -> bool
  }

(* model checker [mc], satisfies mc m ψ ⟺ M ⊨ ψ *)
let mc (m : 'ap tmodel) (psi : 'ap state_prop) : bool =
  let v = mca m psi in
  State.Set.for_all m.s ~f:(fun s ->
    not (m.s0 s) || State.Set.mem v s)
```

```
(* [mca m ψ] is the set of states satisfying ψ *)
let rec mca m psi : State.Set.t =
  let module S = State.Set in
  match psi with
  | True -> m.s
  | False -> S.empty
  | Atom p -> S.filter m.s ~f:(fun s -> m.l s p)
  | Not psi -> S.diff m.s (mca m psi)
  | And (psi1, psi2) ->
    let v1 = mca m psi1
    and v2 = mca m psi2 in
    S.inter v1 v2
  | Or (psi1, psi2) ->
    let v1 = mca m psi1
    and v2 = mca m psi2 in
    S.union v1 v2
  | Impl (psi1, psi2) -> mca m (Or (Not psi1, psi2))
  | A (X psi) ->
    (* A X ψ ≃ ¬E X ¬ψ *)
    mca m (Not (E (X (Not psi))))
  | A (G psi) ->
    (* A G ψ ≃ ¬E F ¬ψ *)
    mca m (Not (E (F (Not psi))))
  | A (F psi) ->
    (* A F ψ ≃ ¬E G ¬ψ *)
    mca m (Not (E (G (Not psi))))
  | A (U (psi1, psi2))
    (* A [ψ1 U ψ2] ≃ ¬[E {¬ψ2 U ¬[ψ1 ∨ ψ2]} ∨ E G ¬ψ2] *)
    mca m (Not (Or
      (E (U (Not psi2, Not (Or (psi1, psi2))))
      , E (G (Not psi2)))))
  | E (X psi) ->
    let v = mca m psi in
    S.filter m.s ~f:(fun s -> S.exists v ~f:(m.t s))
  | E (F psi) -> mca m (E U (True, psi))
  | E (G psi) ->
    (* G and U reason about infinite paths
        (use fixpoint for finite computation) *)
    let v = mca m psi in
    (* Compute initial state set [v], remove states from [v']
        that cannot transition into [v'] *)
    fix v (fun v' ->
      S.filter v' ~f:(fun s -> S.exists v' ~f:(m.t s)) )
  | E (U (psi1, psi2)) ->
    let v1 = mca m psi1
    and v2 = mca m psi2 in
    fix v2 (fun v' ->
      S.union v' (S.filter v1 ~f:(S.exists v' ~f:(m.t s))))
```

Refutations

- **Idea**: $M \not\models^{\mathsf{ACTL}} \psi \iff M \models \neg\psi^{\mathsf{ACTL}}$, by duality $\neg\psi^{\mathsf{ACTL}}$ can be expressed in ECTL (in NNF).
- Use Curry-Howard for '*witnesses*' (terms) with a validity relation (type system).
- Witnesses may be computed using a model checking algorithm (Program synthesis).

$$\mathsf{FinPath}(M, s) = \{\Pi \in \mathsf{List}\ S : \Pi(0) = s \wedge \forall 0 \leq i \leq |\Pi| - 1. \Pi(i) \to \Pi(i+1)\}$$

Syntax

Terms $e ::=$

$$\begin{aligned} &\mid p(s) \\ &\mid \neg p(s) \\ &\mid \langle e, e\rangle \\ &\mid \mathsf{L}\ e \\ &\mid \mathsf{R}\ e \\ &\mid \mathsf{X}(s, s, e) \\ &\mid \mathsf{F}([s, \ldots, s], e) \\ &\mid \mathsf{G}([\langle s, e\rangle, \ldots, \langle s, e\rangle]) \\ &\mid \mathsf{U}([\langle s, e\rangle, \ldots, \langle s, e\rangle], s, e) \end{aligned}$$

Type System

$$\frac{p \in \ell(s)}{s \vdash_M p(s) : p}\ \textsc{Atom} \qquad \frac{p \notin \ell(s)}{s \vdash_M \neg p(s) : \neg p}\ \neg\textsc{Atom} \qquad \frac{s \vdash_M e_1 : \psi_1 \qquad s \vdash_M e_2 : \psi_2}{s \vdash_M \langle e_1, e_2\rangle : \psi_1 \wedge \psi_2}\ \wedge$$

$$\frac{s \vdash_M e : \psi_1}{s \vdash_M \mathsf{L}\ e : \psi_1 \vee \psi_2}\ \vee_1 \qquad \frac{s \vdash_M e : \psi_2}{s \vdash_M \mathsf{R}\ e : \psi_1 \vee \psi_2}\ \vee_2$$

$$\frac{s \to s' \qquad s' \vdash_M e : \psi}{s \vdash_M X(s, s', e) : \mathsf{E}\ \mathsf{X}\ \psi}\ \mathrm{X} \qquad \frac{\Pi \in \mathsf{FinPath}(M, s) \qquad \mathsf{last}(\Pi) \vdash_M e : \psi}{s \vdash_M \mathsf{F}(\Pi, e) : \mathsf{E}\ \mathsf{F}\ \psi}\ \mathrm{F}$$

$$\frac{\Pi = [s_0, \ldots, s_n] \in \mathsf{FinPath}(M, s) \qquad \overbrace{\mathsf{last}(\Pi) \to \Pi(n)}^{\text{lasso-shaped path}} \qquad \forall 0 \leq i \leq n. s_i \vdash_M e_i : \psi}{s \vdash_M \mathsf{G}([\langle s_0, e_0\rangle, \ldots, \langle s_n, e_n\rangle]) : \mathsf{E}\ \mathsf{G}\ \psi}\ \mathrm{G}$$

$$\frac{\begin{array}{c}\Pi = [s_0, \ldots, s_n, s_{n+1}] \in \mathsf{FinPath}(M, s) \\ \forall 0 \leq i \leq n. s_i \vdash_M e_i : \psi_1 \qquad s_{n+1} \vdash_M e_{n+1} : \psi_2\end{array}}{s \vdash_M \mathsf{U}([\langle s_0, e_0\rangle, \ldots, \langle s_n, e_n\rangle, s_{n+1}, e_{n+1}] : \mathsf{E}\ (\psi_1\ \mathsf{U}\ \psi_2))}\ \mathrm{U}$$