model checking using dynamic logic: KeY

specification in Java modeling language
is transformed into formula in dynamic logic
and compared with semantics in dynamic logic

more generally: dynamic logic is useful for sequential programming

see wiki for KeY
see home page of KeY

a conference: advances in modal logic

overview

- propositional dynamic logic: semantics
recap: starting point PDL

for every program \( \alpha \) we have a modality \( \langle \alpha \rangle \)

\( \langle \alpha \rangle \phi \) intuitively means
it is possible to execute \( \alpha \) starting in the current state,
and halt successfully in a state satisfying \( \phi \)

\( [\alpha] \phi \) intuitively means
if \( \alpha \) halts successfully, then it halts in a state satisfying \( \phi \)

program correctness

we restrict attention to input-output behaviour

specification consists of
input condition \( \phi \) and output condition \( \psi \)

partial correctness:
if the program starts satisfying \( \phi \),
and if it halts,
then when it halts \( \psi \) is satisfied

total correctness:
it is partially correct,
and it terminates whenever started satisfying \( \phi \)

remarks

two approaches: exogenous logic and endogenous logic
exogeneous: programs are explicit in the language
propositional dynamic logic (PDL) and Hoare Logic are exogenous
temporal logic and method by Floyd are endogenous
in addition: compositionality for PDL, not for temporal logic
MLOM chapter 14.2

programs and formulas of PDL

we have programs built from atomic programs,
and possibly dependent on propositions

we have formulas built from atomic formulas,
and possibly dependent on programs

truth of \( \phi \) depends on
truth of strictly smaller formulas and relation of strictly smaller programs
consider for example \( \phi = (a)p \)

relation of \( \alpha \) depends on
relation of strictly smaller programs and truth of strictly smaller formulas
consider for example \( \alpha = p?; a \)
non-determinism

we have non-determinism due to choice \( \cup \)
we have non-determinism due to iteration \( \alpha^* \)

semantics of PDL

\( A \): set of atomic program
\( \text{Prog} \): set of regular programs over \( A \)

a PDL-model is a \( \text{Prog} \)-frame which is a PDL-frame with in addition
\( R_{\phi?} = \{(w, w) \mid M, w \models \phi\} \)

we can find a PDL-model as extension of an \( A \)-model with in addition
\( R_{\phi?} = \{(w, w) \mid M, w \models \phi\} \)

exercise from yesterday

\( W = \{s, t, u, v\} \)
\( R_a = \{(t, v), (v, t), (s, u), (u, s)\} \)
\( R_b = \{(u, v), (v, u), (s, t), (t, s)\} \)
\( V(p) = \{u, v\} \)
\( V(q) = \{t, v\} \)

we have \( p \leftrightarrow [(ab^*a)^*]p \)
we have \( q \leftrightarrow [(ba^*b)^*]q \)

example

\( W = \{u, v, w\} \)
\( R_a = \{(u, v), (u, w), (v, w), (w, v)\} \)
\( V(p) = \{u, v\} \)

we have \( u \models \langle a \rangle \neg p \land \langle a \rangle p \)
we have \( v \models [a] \neg p \)
we have \( w \models [a] p \)

in every world (state) we have \( \langle a^* \rangle [(aa)^*]p \land \langle a^* \rangle [(aa)^*] \neg p \)
which of the two directions is valid in PDL?

\[(a \cup b)^* p \leftrightarrow [a^*]p \land [b^*]p\]

**while**

**while** \(p \text{ do } a \) encoded as \((p?; a)^*; \neg p\)?

**formula** \((\text{while } p \text{ do } a) \ q\) is valid in model \(M\) in state \(x\) iff

there exist \(n \geq 0\) and there exist \(x_0, \ldots, x_n\) such that

\[x = x_0, \text{ and} \]
\[M, x \models p \text{ and } x = x_0 \text{ and } R_a x_0 x_1 \]
\[M, x_0 \models p \text{ and } R_a x_0 x_1 \]
\[\vdots \]
\[M, x_{n-1} \models p \text{ and } R_a x_{n-1} x_n \]
\[M, x_n \not\models p \text{ and } M, x_n \models q\]

**back to the approach due to Hoare**

we encode while-programs as regular programs

we encode \(\{\phi\} P \{\psi\}\) as \(\phi \rightarrow [Q] \psi\) with \(Q\) the translation of \(P\)

we show that all rules from Hoare Logic are derivable

so we can encode Hoare logic in propositional dynamic logic