program correctness

Hoare Logic is used to prove program correctness of while programs

Hoare Logic uses preconditions and postconditions: \( \{ \phi \} P \{ \psi \} \)

we use regular programs which slightly generalize while programs

we use Propositional Dynamic Logic to express correctness

overview

propositional dynamic logic (PDL): starting point

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 \( \phi \)
set Prog of PDL (or regular) programs: definition

atomic program
  a from a set $A$ of atomic programs
sequential composition
  $\alpha; \beta$
non-deterministic choice
  $\alpha \cup \beta$
iteration
  $\alpha^*$
test
  $\phi?$ with $\phi$ a formula, so depends on the grammar for formulas

PDL programs: intuitive meaning

$\alpha$
  atomic, indecomposable, step
$\phi?$
  if $\phi$ then skip else abort, that is,
  if $\phi$ holds then continue without changing state,
  if $\phi$ does not hold then block without halting
$\alpha; \beta$
  do $\alpha$, then do $\beta$
$\alpha \cup \beta$
  choose $\alpha$ or $\beta$ and execute it
$\alpha^*$
  choose $n \geq 0$ and execute $\alpha$ $n$ times

PDL formulas: definition

atomic formula
  $p$ from a set $\text{Var}$ of atomic propositions
true
  $\top$
negation
  $\neg \phi$
conjunction
  $\phi \land \psi$
diamond
  $\langle \alpha \rangle \phi$, with $\alpha$ a program, so depends on the grammar for programs

mutual dependency: examples

$[p]?p$
  if $?p$ halts then in a state satisfying $p$
$\langle p? \rangle p$
  it is possible to execute $p?$ and halt in a state where $p$ holds
$[\alpha] \bot$
  $\alpha$ never terminates
$[\alpha] \top$
  is always true
$\top?$
  is skip
$\bot?$
  is fail
PDL formulas: examples

\([\alpha \cup \beta] \phi\)
always if we execute \(\alpha\) or \(\beta\) we arrive at a state where \(\phi\) holds

\(((\alpha \beta)^*) \phi\)
there is a sequence of alternating executions of \(\alpha\) and \(\beta\) bringing us to a state where \(\phi\) holds

\(\langle \alpha^* \rangle \phi \leftrightarrow \phi \lor \langle \alpha; \alpha^* \rangle \phi\)
\(\phi\) holds after a finite number of \(\alpha\) steps
if and only if
either \(\phi\) holds here, or we can do an \(\alpha\) step and then more \(\alpha\) steps to reach
a state where \(\phi\) holds

while programs and regular programs

while programs can be encoded as regular programs:

\[
\text{if } \phi \text{ then } \alpha \text{ else } \beta = (\phi ?; \alpha) \cup (\neg \phi ?; \beta)
\]

\[
\text{while } \phi \text{ do } \alpha = (\phi ?; \alpha^*; \neg \phi ?)
\]

towards a semantics for PDL formulas

we obtain the semantics as an instance of multi-modal logic
in particular:
\[
\mathcal{M}, s \models \langle \alpha \rangle \phi \text{ iff there is } s' \text{ such that } (s, s') \in R_\alpha \text{ and } \mathcal{M}, s' \models \phi
\]
however:
an arbitrary model does respect the intended meaning of the programs
therefore we will impose conditions on the relations \(R_\alpha\)

alternative approach in MLOM 14.3

PDL formulas: more examples

\([\alpha] \phi \land \psi \leftrightarrow [\alpha] \phi \land [\alpha] \psi\) (seems a tautology)

\([\alpha; \beta] \phi \leftrightarrow [\alpha][\beta] \phi\) (seems a tautology)

\([\alpha] p \leftrightarrow [\beta] p\) (gives an equivalence between \(\alpha\) and \(\beta\)
intuitive requirements for a PDL model

consider \( a; b \) and \( R_{a;b} \)

consider \( a \cup b \) and \( R_{a\cup b} \)

consider \( a^* \) and \( R_{a^*} \)

this suggests to start from all the \( R_a \) with \( a \in A \) an atomic program

but what to do with \( R_{\phi} \) ?

PDL frame: definition

a Prog-frame \( \mathcal{F} = (W, \{ R_\alpha | \alpha \in \text{Prog} \}) \) is a PDL-frame if

\[
R_{\alpha \beta} = R_\alpha \circ R_\beta, \quad \text{and} \\
R_{\alpha \cup \beta} = R_\alpha \cup R_\beta, \quad \text{and} \\
R_{\alpha^*} = (R_\alpha)^*
\]

so if we know all \( R_a \) then we know enough!

what are the definitions on the relations?

more definitions on relations

the identity relation: \( \text{Id} = \{(x, x)\} \)

the \( n \)-fold composition of \( R \): \( R^0 = \text{Id} \) and \( R^{n+1} = R^n \circ R \)

the reflexive-transitive closure of \( R \): \( R^* = \bigcup_{n \geq 0} R^n \)

note: if \( x R^* y \), then there exists \( n \geq 0 \) and there exist \( x_1, \ldots, x_{n-1} \) such that \( x = x_0 R x_1 R \ldots R x_n = y \)

note: \( R^* \) is the smallest reflexive and transitive relation containing \( R \)

definitions on relations

the composition of \( R \) and \( S \): \( R \circ S = \{(x, z) | \exists y : Rxy \land Syz\} \)

the union of \( R \) and \( S \): \( R \cup S = \{(x, y) | Rxy \lor Sxy\} \)
PDL model: definition

a model $\mathcal{M} = (W, \{ R_\alpha | \alpha \in \text{Prog} \}, V)$ is a PDL-model if

$(W, \{ R_\alpha | \alpha \in \text{Prog} \}$ is a PDL-frame, and

$R_\phi = \{(w, w) | \mathcal{M}, w \models \phi \}$

PDL extension: definition

now it is sufficient to know the $R_a$ for all $a \in A$

Let $\mathcal{M} = (W, \{ R_a | a \in A \}, V)$ be an $A$-model

Its PDL-extension is defined as $\hat{\mathcal{M}} = (W, \{ \hat{R}_\alpha | \alpha \in \text{Prog} \}, V)$ with

$\hat{R}_a = R_a$

$\hat{R}_{\alpha \beta} = \hat{R}_\alpha \circ \hat{R}_\beta$

$\hat{R}_{a \cup \beta} = \hat{R}_a \cup \hat{R}_\beta$

$\hat{R}_\phi = \{(x, x) | \mathcal{M}, x \models \phi \}$

can we really encode Hoare Logic?

we encode a while-program as a regular program

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: yes)