overview

- multi-modal logic

- towards propositional dynamic logic

---

multi-modal logic: formulas

We assume a set of labels $I$

For every label $i$ there is a modality $\langle i \rangle$

So the formulas of multi-modal logic are, given $I$, inductively defined by:

$p, \perp, \neg \phi, \phi \land \psi, \phi \lor \psi, \langle i \rangle \phi, [i] \phi$ for $i \in I$
multi-modal logic: frames and models

let $\mathcal{I}$ be a set of indices or labels

an $\mathcal{I}$-frame is a pair $(W, \{R_i | i \in \mathcal{I}\})$ with

$W \neq \emptyset$ a set of worlds or states
$R_i \subseteq W \times W$ for every $i \in \mathcal{I}$

an $\mathcal{I}$-model is a triple $(W, \{R_i | i \in \mathcal{I}\}, V)$ with

$(W, \{R_i | i \in \mathcal{I}\})$ an $\mathcal{I}$-frame and $V : \text{Var} \to \mathcal{P}(W)$ a valuation

example

use index set $\mathcal{I} = \{a, b, c\}$
give a model with a world where the formula

$\langle a \rangle \langle b \rangle [a] p \land [c] \neg \langle a \rangle p$ is true

multi-modal logic: truth and validity

let $\mathcal{M} = (W, \{R_i | i \in \mathcal{I}\}, V)$ be an $\mathcal{I}$-model

$\mathcal{M}, w \models \phi$ is defined by induction on the definition of formulas

important clauses:

$\mathcal{M}, w \models \langle \alpha \rangle \phi$ iff $\mathcal{M}, v \models \phi$ for some $v$ with $R_\alpha w v$

$\mathcal{M}, w \models [\alpha] \phi$ iff $\mathcal{M}, v \models \phi$ for all $v$ with $R_\alpha w v$

truth in a model, valid in a frame, universal validity are defined as before

multi-modal logic: bisimulation

we use one index set $\mathcal{I}$

let $\mathcal{M} = (W, \{R_i | i \in \mathcal{I}\}, V)$ and $\mathcal{M}' = (W', \{R'_i | i \in \mathcal{I}\}, V')$ be $\mathcal{I}$-models

$\emptyset \neq Z \subseteq W \times W'$ is a bisimulation if for every $(w, w') \in Z$

$w \in V(p)$ if and only if $w' \in V'(p)$

if $R_i w v$ then there is $v'$ with $R'_i w v'$ and $(v, v') \in Z$ (same $i$!)

if $R'_i w' v'$ then there is $v$ with $R_i w v$ and $(v, v') \in Z$ (same $i$!)
theorem: modal equivalence and bisimulation

under the restriction that every $R_i$ is finitely branching

(which does not imply that the model is finitely branching)

two worlds are bisimilar if and only if they are modally equivalent

theorem by Hennessy and Milner (1985)

instances of modal logic

we already know various instances of multi-modal logic:

$I = \emptyset$ gives propositional logic

$I = \{0\}$ gives basic modal logic

$I = \{F, P\}$ gives temporal logic provided $R_F = R_{F}^{-1}$

and we will see $I = \text{Prog}(A)$ gives propositional dynamic logic

multi-modal logic: book

MLOM chapter 10.1

especially interesting if there is some connection between the $R_i$ such as the requirement for temporal logic that past is the inverse of future

basic temporal logic as instance of multi-modal logic

we take as set of label $I = \{F, P\}$

and we require $R_P s t$ if and only if $R_F t s$ for all states $s$ and $t$

the requirement can be enforced using modal logic:

$F \models q \rightarrow [P](F) q$ if and only if $R_P \subseteq R_F^{-1}$

and for the other direction or inclusion:

$F \models q \rightarrow [F](P) q$ if and only if $R_F \subseteq R_P^{-1}$

multi-modal logic: book

MLOM chapter 10.1
overview

- multi-modal logic
- towards propositional dynamic logic

program verification

prove that a program meets its specification

for example:

input: finite list of integers
program: sort
output: sorted permutation of the input

example: gcd program

```plaintext
while y ≠ 0 do
  begin
    z := x mod y;
    x := y;
    y := z;
  end
return x
```

x is an input variable and an output variable
y is an input variable
z is a work variable

example: run of gcd program

a trace or a run is a sequence of states in the gcd example:

a state is a tuple of values of x, y, z; example of a run:

(15, 27, 0) → (15, 27, 15) → (27, 27, 15) → (27, 15, 15) →
(27, 15, 12) → (15, 15, 12) → (15, 12, 12) → (15, 12, 3) →
(12, 12, 3) → (12, 3, 3) → (3, 3, 0) → (3, 0, 0)
example: correctness of gcd program

pre:

\[ x, y \in \{0, 1, 2, \ldots \} \text{ and } y \neq 0 \land c = x \land d = y \]

during:

\[ \text{gcd}(x, y) = \text{gcd}(c, d) \text{ is an invariant} \]

post:

\[ y = 0 \land x = \text{gcd}(c, d) \]

program verification: approach by Hoare

prove statements of the form \{precondition\}program\{postcondition\}

the precondition and postcondition are formulas

the program is a while-program

built from sequential composition, conditional, while, assignment

we have proof rules for showing \{\phi\} \alpha \{\psi\}

Tony Hoare

Turing Award 1980

quicksort,

Hoare Logic following work by Floyd and Turing,

Communicating Sequential Processes (CSP)

Alan Turing

the work on program verification by Tony Hoare

actually similar to work by Turing
towards program verification using modal logic

a state of a program execution is a state or world

a program is a regular program which slightly generalizes while program

a statement \{pre\}program\{post\} is a formula pre → [program]post

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 \(\phi\)