Proof-based verification in Event-B

Michael Butler

SETSS 2016, Chongqing

Validation and verification

- Requirements validation:
  - The extent to which (informal) requirements satisfy the needs of the stakeholders
- Model validation:
  - The extent to which (formal) model accurately captures the (informal) requirements
- Model verification:
  - The extent to which a model correctly maintains invariants or refines another (more abstract) model
- Code verification:
  - The extent to which a program correctly implements a specification/model

Verification through Proof Obligations

- Proof obligations (PO) are mathematical theorems derived from a formal model (or program)

- The validity of a PO is proved using deductive rules of logic and set theory, e.g.,

$$S \subseteq T \iff (\forall x \cdot x \in S \Rightarrow x \in T)$$

$$x \in (S \cup T) \iff (x \in S \vee x \in T)$$

Invariant Preservation PO

- Assume: variables v and invariant Inv (v is free in Inv)
- Event:

  Ev ≙ **any** x **where** Grd **then** v := Exp **end**

- PO to prove Ev preserves Inv: prove that the following sequent is valid:

INV PO:	Inv, Grd ⊢ Inv[v:=Exp]

That is, prove that the updated invariant, Inv[v:=Exp] , follows from the invariant, Inv, and the guard, Grd

Sequent

- A sequent consists of Hypotheses (H) and a Goal (G), written

  $H \vdash G$

- A sequent is valid if G follows from H

- Event-B proof obligations (PO) are sequents

  Assumptions $\vdash$ Goal

Substitution

- Replace all *free* occurrence of variable x by expression E in predicate P:

$$P\ [\ x := E\ ]$$

- Example:

$$(\ 0<n\ \wedge\ n\leq 10\ )\ [n:=7] \quad \Leftrightarrow \quad 0<7\ \wedge\ 7\leq 10$$

- Bound variables are quantified variables:

$$(\ \forall n \bullet n>0 \Rightarrow 1\leq n\ )\ [n:=7] \quad \Leftrightarrow \quad (\ \forall n \bullet n>0 \Rightarrow 1\leq n\ )$$

  Here n is bound in the predicate so is not substituted

- Free variables are variables that appear in P that are not *bound* within P.

Multiple Substitution

$Q\,[\,x_1,x_2,\ldots,x_n := E_1,E_2,\ldots,E_n\,]$

- Examples:

$(l<n \wedge n\leq m)\,[\,l,m,n:=0,10,7\,]$

$\Leftrightarrow\; 0<7 \wedge 7\leq 10$

$(\,in \cap out = \{\}\,)\,[\,in, out := in\backslash\{u\}, out \cup \{u\}\,]$

$\Leftrightarrow\; ?$

Multiple Substitution

$Q\ [\ x_1,x_2,\ldots,x_n := E_1,E_2,\ldots,E_n\ ]$

- Examples:

$(l<n \wedge n\leq m)\ [\ l,m,n:=0,10,7\ ]$

$\Leftrightarrow\ 0<7 \wedge 7\leq 10$

$(\ in \cap out = \{\}\ )\ [\ in, out := in\backslash\{u\}, out \cup \{u\}\ ]$

$\Leftrightarrow\ (in\backslash\{u\}) \cap (out \cup \{u\}) = \{\}$

Example Invariant Preservation PO

INV PO rule: $Inv,\ Grd \vdash Inv[v := Exp]$

Example:

- Invariant: $x + y = C$
- Event: $x, y := x + 1, y - 1$

PO for example:

$$x + y = C \vdash (x+1) + (y-1) = C$$

Rodin demo

- Proof obligations
- Restrict capacity of building

Model Checking versus Deductive Proof

- Model checking: force the model to be finite state and explore state space looking for invariant violations
  - completely automatic
  - powerful debugging tool (counter-example)

- (Semi-)automated proof: based on logical deduction rules
  - no restrictions on state space
  - leads to discovery of invariants that deepen understanding
  - not completely automatic