Summary of Event-B Proof Obligations

Jean-Raymond Abrial
(edited by Thai Son Hoang)

Department of Computer Science
Swiss Federal Institute of Technology Zürich (ETH Zürich)

Bucharest DEPLOY 2-day Course, 14th-16th, July, 2010

Purpose of this Presentation

- Prerequisite:
  1. Summary of Mathematical Notation (a quick review)
  2. Summary of Event-B Notation

  Examples developed in (2) will be used here
- Showing the various Event-B proof obligations (sometimes also called verification conditions)

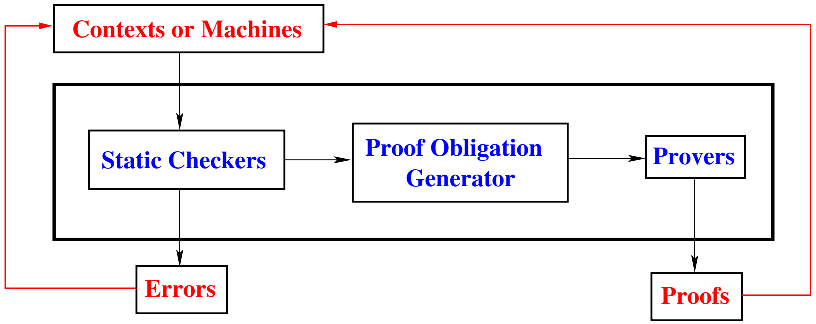
Role of the Proof Obligation Generator

- The POs are automatically generated by a Rodin Platform tool called the Proof Obligation Generator
- This tool is run after the Static Checker (which static checks contexts or machine texts)
- The Proof Obligation Generator decides then what is to be proved
- The outcome are various sequents, which are transmitted to the provers performing automatic or interactive proofs

Summary of the Main Rodin Platform Kernel Tools

- The Static Checkers:
  - lexical analyser
  - syntactic analyser
  - type checker
- The Proof Obligation Generator
- The Provers

Summary of the Main Rodin Platform Kernel Tools



- Proofs which cannot be done help improving the model

Various Kinds of Proof Obligations

- Invariant preservation (initial model) (INV slide 9)
- Non-deterministic action feasibility (FIS slide 14)
- Guard strengthening in a refinement (GRD slide 18)
- Invariant preservation in a refinement (INV slide 22)
- Simulation (SIM slide 26)
- Numeric variant (NAT slide 30)
- Set variant (FIN slide 34)

Various Kinds of Proof Obligations (cont'd)

- Variant decreasing (VAR slide 38)
- Feasibility of a non-deterministic witness (WFIS slide 46)
- Proving theorems (THM slide 50)
- Well-definedness (WD slide 58)
- Guard strengthening when merging abstract events (MRG slide 62)

Outline of each Proof Obligation

- Purpose and naming
- Formal definition
- Where generated in the "search" example
- Application to the example

Purpose of Invariant Preservation PO (INV) (for Initial Model)

- Ensuring that each invariant is preserved by each event.
- For an event "evt" and an invariant "inv" the name of this PO is:

  evt/inv/INV

Formal Definition of Invariant Preservation (INV) (for Initial Model)

```
evt
  any x where
    G(x, s, c, v)
  then
    v :| BAP(x, s, c, v')
  end
```

s	:	seen sets
c	:	seen constants
v	:	variables
$A(s,c)$	:	seen axioms
$I(s,c,v)$	:	invariants
evt	:	specific event
x	:	event parameters
$G(x,s,c,v)$	:	event guards
$BAP(x,s,c,v')$	:	event before-after predicate
$i(s,c,v')$	:	modified specific invariant

Axioms Invariants Guards of the event Before-after predicate of the event $\vdash$ Modified Specific Invariant	$evt\,/\,inv\,/$INV

$A(s,c)$
$I(s,c,v)$
$G(x,s,c,v)$
$BAP(x,s,c,v')$
$\vdash$
$i(s,c,v')$

- In case of the initialization event, $I(s,c,v)$ is removed from the hypotheses

Examples in Machine **m_0a** (INV)

```
context
  ctx_0
sets
  D
constants
  n
  f
  v
axioms
  axm1 : n ∈ ℕ
  axm2 : f ∈ 1..n → D
  axm3 : v ∈ ran(f)
  thm1 : n ∈ ℕ1
end
```

```
machine
  m_0a
sees
  ctx_0
variables
  i
invariants
  inv1 : i ∈ 1 .. n
events
  . . .
end
```

```
initialisation ≙
  status
    ordinary
  then
    act1 : i := 1
  end
```

```
search ≙
  status
    ordinary
  any
    k
  where
    grd1 : k ∈ 1 .. n
    grd2 : f(k) = v
  then
    act1 : i := k
  end
```

- Two invariant preservation POs are generated:
  - **initialisation**/**inv1**/INV
  - **search**/**inv1**/INV

Proof Obligation initialisation/inv1/INV

axm1 **axm2** **axm3** **thm1** BA predicate $\vdash$ modified **inv1**	$n \in \mathbb{N}$ $f \in 1\,..\,n \rightarrow D$ $v \in \mathrm{ran}(f)$ $n \in \mathbb{N}1$ $i' = 1$ $\vdash$ $i' \in 1\,..\,n$	$n \in \mathbb{N}$ $f \in 1\,..\,n \rightarrow D$ $v \in \mathrm{ran}(f)$ $n \in \mathbb{N}1$ $\vdash$ $1 \in 1\,..\,n$

Simplification performed by the PO Generator

```
initialisation ≙
  status
    ordinary
  then
    act1 : i := 1
  end
```

- Note that **inv1** is not part of the hypotheses (we are in the **initialisation** event)

Proof Obligation search/inv1/INV

axm1 axm2 axm3 thm1 inv1 grd1 grd2 BA predicate $\vdash$ modified **inv1**	$n \in \mathbb{N}$ $f \in 1\,..\,n \to D$ $v \in \mathrm{ran}(f)$ $n \in \mathbb{N}1$ $i \in 1\,..\,n$ $k \in 1\,..\,n$ $f(k) = v$ $i' = k$ $\vdash$ $i' \in 1\,..\,n$	$n \in \mathbb{N}$ $f \in 1\,..\,n \to D$ $v \in \mathrm{ran}(f)$ $n \in \mathbb{N}1$ $i \in 1\,..\,n$ $k \in 1\,..\,n$ $f(k) = v$ $\vdash$ $k \in 1\,..\,n$

Simplification performed by the PO Generator

```
search  ≙
  status
    ordinary
  any
    k
  where
    grd1 : k ∈ 1 .. n
    grd2 : f(k) = v
  then
    act1 :  i := k
  end
```

- In what follows, we'll show the simplified form only

Purpose of the Feasibility PO (FIS)

- Ensuring that each non-deterministic action is feasible.
- For an event "evt" and a non-deterministic action "act" in it, the name of this PO is:

  evt/act/FIS

Formal Definition of the Feasibility PO (FIS)

```
evt
  any  x  where
    G(x, s, c, v)
  then
    v :| BAP(x, s, c, v, v')
  end
```

s	:	seen sets
c	:	seen constants
v	:	variables
$A(s,c)$	:	seen axioms
$I(s,c,v)$	:	invariants
evt	:	specific event
x	:	event parameters
$G(x,s,c,v)$	:	event guards
$BAP(x,s,c,v,v')$	:	event action

Axioms Invariants Guards of the event $\vdash$ $\exists v' \cdot$ Before-after predicate	$evt/act/$FIS

$A(s,c)$
$I(s,c,v)$
$G(x,s,c,v)$
$\vdash$
$\exists v' \cdot BAP(x,s,c,v,v')$

Example in Machine m_0b (FIS)

```
context
  ctx_0
sets
  D
constants
  n
  f
  v
axioms
  axm1 : n ∈ ℕ
  axm2 : f ∈ 1..n → D
  axm3 : v ∈ ran(f)
  thm1 : n ∈ ℕ1
end
```

```
machine
  m_0b
sees
  ctx_0
variables
  i
invariants
  inv1 :  i ∈ 1 .. n
events
  . . .
end
```

```
initialisation  ≙
  status
    ordinary
  then
    act1 :  i := 1
  end
```

```
search  ≙
  status
    ordinary
  then
    act1 :  i :| i' ∈ 1 .. n ∧ f(i') = v
  end
```

- Among others, one feasibility PO is generated:
  - **search/act1**/FIS

Proof Obligation search/act1/FIS

axm1
axm2
axm3
thm1
inv1
grd
$\vdash$
$\exists i' \cdot$ before-after predicate

$n \in \mathbb{N}$
$f \in 1\,..\,n \to D$
$v \in \mathrm{ran}(f)$
$n \in \mathbb{N}1$
$i \in 1\,..\,n$
no guard in event **search**
$\vdash$
$\exists i' \cdot i' \in 1\,..\,n \;\wedge\; f(i') = v$

search $\widehat{=}$
  status
    ordinary
  then
    act1 : $i :\mid i' \in 1\,..\,n \;\wedge\; f(i') = v$
  end

Purpose of the Guard Strengthening PO (GRD)

- Ensuring that the concrete guards in the refining event are stronger than the abstract ones.
- This ensures that when a concrete event is enabled then so is the corresponding abstract one.
- For a concrete event "evt" and an abstract guard "grd" in the corresponding abstract event, the name of this PO is:

  evt/grd/GRD

Formal Def. of the Guard Strengthening PO (GRD)

evt0
  any
    x
  where
    $g(x, s, c, v)$
    ...
  then
    ...
  end

evt
  refines
    evt0
  any
    y
  where
    $H(y, s, c, w)$
  with
    $x : W(x, y, s, c, w)$
  then
    ...
  end

s	:	seen sets
c	:	seen constants
v	:	abstract variables
w	:	concrete variables
$A(s, c)$	:	seen axioms
$I(s, c, v)$	:	abs. invts.
$J(s, c, v, w)$	:	conc. invts.
evt	:	specific concrete event
x	:	abstract event parameter
y	:	concrete event parameter
$g(x, s, c, v)$	:	abstract event specific guard
$H(y, s, c, w)$	:	concrete event guards

Axioms Abstract invariants Concrete invariants Concrete event guards witness predicate $\vdash$ Abstract event specific guard	*evt / grd /* GRD

$A(s, c)$
$I(s, c, v)$
$J(s, c, v, w)$
$H(y, s, c, w)$
$W(x, y, s, c, w)$
$\vdash$
$g(x, s, c, v)$

- It is simplified when there are no parameters

Example in Mch m_1a Refining Mch m_0a (GRD)

machine
  m_1a
refines
  m_0a
sees
  ctx_0
variables
  i
  j
invariants
  inv1 : $j \in 0\,..\,n-1$
  inv2 : $v \notin f[1\,..\,j]$
  thm1 : $v \in f[j+1\,..\,n]$
variant
  $n - j$
events
  ...
end

initialisation $\widehat{=}$
  status **ordinary**
  then
    act1 : $i := 1$
    act2 : $j := 0$
  end

search $\widehat{=}$
  status **ordinary**
  refines
    search
  when
    grd1 : $f(j+1) = v$
  with
    k : $j + 1 = k$
  then
    act1 : $i := j + 1$
  end

(abstract-)**search** $\widehat{=}$
  status **ordinary**
  any
    k
  where
    grd1 : $k \in 1\,..\,n$
    grd2 : $f(k) = v$
  then
    act1 : $i := k$
  end

progress $\widehat{=}$
  status **convergent**
  when
    grd1 : $f(j+1) \neq v$
  then
    act1 : $j := j + 1$
  end

- Among others, two guard strengthening POs are generated:
  - **search**/**grd1**/GRD
  - **search**/**grd2**/GRD

Proof Obligation search/grd2/GRD

axm1
axm2
axm3
thm1 of **ctx_0**
inv1 (abstract)
inv1 (concrete)
inv2 (concrete)
thm1 of **m_1a**
grd1 (concrete)
witness predicate
$\vdash$
grd2 (abstract)

$n \in \mathbb{N}$
$f \in 1\,..\,n \rightarrow D$
$v \in \mathrm{ran}(f)$
$n \in \mathbb{N}1$
$i \in 1\,..\,n$
$j \in 0\,..\,n-1$
$v \notin f[1\,..\,j]$
$v \in f[j+1\,..\,n]$
$f(j+1) = v$
$j+1 = k$
$\vdash$
$f(k) = v$

search $\widehat{=}$
status
ordinary
refines
search
when
grd1 : $f(j+1) = v$
with
k : $j+1 = k$
then
act1 : $i := j+1$
end

(abstract-)**search** $\widehat{=}$
status
ordinary
any
k
where
grd1 : $k \in 1\,..\,n$
grd2 : $f(k) = v$
then
act1 : $i := k$
end

Purpose of Invariant Preservation PO (INV) (for a Refinement)

- Ensuring that each concrete invariant is preserved by each pair of concrete and abstract events.
- For an event "evt" and a concrete invariant "inv" the name of this PO is:

evt/inv/INV

Formal Definition of Invariant Preservation (INV) (for a Refinement)

evt0
any
x
where
...
then
$v :| BA1(v, v', \ldots)$
end

evt
refines
evt0
any
y
where
$H(y, s, c, w)$
with
$x : W1(x, y, s, c, w)$
$v' : W2(y, v', s, c, w)$
then
$w :| BA2(w, w', \ldots)$
end

s	:	seen sets
c	:	seen constants
v	:	abstract vrbls
w	:	concrete vrbls
$A(s, c)$	:	seen axioms
$I(s, c, v)$	:	abs. invts.
$J(s, c, v, w)$	:	conc. invts.
evt	:	concrete event
x	:	abstract prm
y	:	concrete prm
$H(y, s, c, w)$	:	concrete guards
$BA2(w, w', \ldots)$	:	abstract action
$j(s, c, v', w')$	:	modified specific invariant

Axioms Abstract invariants Concrete invariants Concrete event guards witness predicate witness predicate Concrete before-after predicate $\vdash$ Modified Specific Invariant	*evt* / *act* / SIM

$A(s, c)$
$I(s, c, v)$
$J(s, c, v, w)$
$H(y, s, c, w)$
$W1(x, y, s, c, w)$
$W2(y, v', s, c, w)$
$BA2(w, w', \ldots)$
$\vdash$
$j(s, c, v', w')$

- In case of the initialization event, $I(s, c, v)$ and $J(s, c, v, w)$ is removed from the hypotheses

Example in Mch m_1a Refining Mch m_0a (INV)

machine
m_1a
refines
m_0a
sees
ctx_0
variables
i
j
invariants
inv1 : $j \in 0\,..\,n-1$
inv2 : $v \notin f[1\,..\,j]$
thm1 : $v \in f[j+1\,..\,n]$
variant
$n - j$
events
...
end

initialisation $\widehat{=}$
status ordinary
then
act1 : $i := 1$
act2 : $j := 0$
end

search $\widehat{=}$
status ordinary
refines
search
when
grd1 : $f(j+1) = v$
with
k : $j+1 = k$
then
act1 : $i := j+1$
end

(abstract-)**search** $\widehat{=}$
status ordinary
any
k
where
grd1 : $k \in 1\,..\,n$
grd2 : $f(k) = v$
then
act1 : $i := k$
end

- Among others, four invariant preservation POs are generated:
  - **progress/inv1**/INV
  - **progress/inv2**/INV
  - **initialization/inv1**/INV
  - **initialization/inv2**/INV

progress $\widehat{=}$
status convergent
when
grd1 : $f(j+1) \neq v$
then
act1 : $j := j+1$
end

Proof Obligation progress/inv1/INV

axm1
axm2
axm3
thm1 of **ctx_0**
inv1 (abstract)
inv1 (concrete)
inv2 (concrete)
thm1 of **m_1a**
grd1 (concrete)
$\vdash$
modified specific invariant

$n \in \mathbb{N}$
$f \in 1 \,..\, n \rightarrow D$
$v \in \mathrm{ran}(f)$
$n \in \mathbb{N}1$
$i \in 1 \,..\, n$
$j \in 0 \,..\, n-1$
$v \notin f[1 \,..\, j]$
$v \in f[j+1 \,..\, n]$
$f(j+1) \neq v$
$\vdash$
$j+1 \in 0 \,..\, n-1$

progress $\widehat{=}$
status **convergent**
when
grd1 : $f(j+1) \neq v$
then
act1 : $j := j+1$
end

Purpose of the Simulation PO (SIM)

- Ensuring that each action in a concrete event simulates the corresponding abstract action
- This ensures that when a concrete event is "executed" then what it does is not contradictory with what the corresponding abstract event does.
- For a concrete event "evt" and an action "act" in abstract event, the name of this PO is:

  evt/act/SIM

Formal Definition of the Simulation PO (SIM)

evt0
any
x
where
. . .
then
$v :| \; BA1(v, v', \ldots)$
end

evt
refines
evt0
any
y
where
$H(y, s, c, w)$
with
$x : W1(x, y, s, c, w)$
$v' : W2(y, v', s, c, w)$
then
$w :| \; BA2(w, w', \ldots)$
end

s	:	seen sets
c	:	seen constants
v	:	abstract vrbls
w	:	concrete vrbls
$A(s, c)$	:	seen axioms
$I(s, c, v)$	:	abs. invts.
$J(s, c, v, w)$	:	conc. invts.
evt	:	concrete event
x	:	abstract prm
y	:	concrete prm
$H(y, s, c, w)$	:	concrete guards
$BA1(v, v')$	:	abstract action
$BA2(w, w')$	:	concrete action

Axioms Abstract invariants Concrete invariants Concrete event guards witness predicate witness predicate Concrete before-after predicate $\vdash$ Abstract before-after predicate	$evt\,/\,act\,/\,$SIM

$A(s, c)$
$I(s, c, v)$
$J(s, c, v, w)$
$H(y, s, c, w)$
$W1(x, y, s, c, w)$
$W2(y, v', s, c, w)$
$BA2(w, w', \ldots)$
$\vdash$
$BA1(v, v', \ldots)$

Example in Mch m_1a Refining Mch m_0a (SIM)

machine
m_1a
refines
m_0a
sees
ctx_0
variables
i
j
invariants
inv1 : $j \in 0 \,..\, n-1$
inv2 : $v \notin f[1 \,..\, j]$
thm1 : $v \in f[j+1 \,..\, n]$
variant
$n - j$
events
. . .
end

initialisation $\widehat{=}$
status **ordinary**
then
act1 : $i := 1$
act2 : $j := 0$
end

search $\widehat{=}$
status **ordinary**
refines
search
when
grd1 : $f(j+1) = v$
with
k : $j+1 = k$
then
act1 : $i := j+1$
end

(abstract-)**search** $\widehat{=}$
status
ordinary
any
k
where
grd1 : $k \in 1 \,..\, n$
grd2 : $f(k) = v$
then
act1 : $i := k$
end

progress $\widehat{=}$
status **convergent**
when
grd1 : $f(j+1) \neq v$
then
act1 : $j := j+1$
end

- Among others, one simulation PO is generated:
  - **search**/**act1**/SIM

Proof Obligation search/act1/SIM

axm1	$n \in \mathbb{N}$
axm2	$f \in 1\,..\,n \rightarrow D$
axm3	$v \in \mathrm{ran}(f)$
thm1 of **ctx_0**	$n \in \mathbb{N}1$
inv1 (abstract)	$i \in 1\,..\,n$
inv1 (concrete)	$j \in 0\,..\,n-1$
inv2 (concrete)	$v \notin f[1\,..\,j]$
thm1 of **m_1a**	$v \in f[j+1\,..\,n]$
grd1 (concrete)	$f(j+1) = v$
witness predicate	$j+1 = k$
$\vdash$	$\vdash$
before-after predicate (abstract)	$k = j+1$

search $\widehat{=}$
  status
    ordinary
  refines
    search
  when
    grd1 : $f(j+1) = v$
  with
    k : $j+1 = k$
  then
    act1 : $i := j+1$
  end

(abstract-)**search** $\widehat{=}$
  status
    ordinary
  any
    k
  where
    grd1 : $k \in 1\,..\,n$
    grd2 : $f(k) = v$
  then
    act1 : $i := k$
  end

Purpose of the Numeric Variant PO (NAT)

- Ensuring that under the guards of each convergent event a proposed numeric variant is indeed a natural number
- For a convergent event "evt", the name of this PO is:

  evt/NAT

Formal Definition of the Numeric Variant PO (NAT)

machine
  m
refines
  ...
sees
  ...
variables
  v
invariants
  $I(s, c, v)$
events
  ...
variant
  $n(s, c, v)$
end

evt
  status
    convergent
  any x **where**
    $G(x, s, c, v)$
  then
    A
  end

s	:	seen sets
c	:	seen constants
v	:	variables
$A(s, c)$	:	seen axioms
$I(s, c, v)$	:	abs. invts.
$J(s, c, v, w)$	:	conc. invts.
evt	:	specific event
x	:	event parameters
$G(x, s, c, v)$	:	event guards
$n(s, c, v)$	:	numeric variant

Axioms and theorems Abstract invariants and theorems Concrete invariants and theorems Event guards $\vdash$ a numeric variant is a natural number	evt/NAT	$A(s, c)$ $I(s, c, v)$ $J(s, c, v, w)$ $G(x, s, c, v)$ $\vdash$ $n(s, c, v) \in \mathbb{N}$

Example in Mch m_1a Refining Mch m_0a (NAT)

machine
  m_1a
refines
  m_0a
sees
  ctx_0
variables
  i
  j
invariants
  inv1 : $j \in 0\,..\,n-1$
  inv2 : $v \notin f[1\,..\,j]$
  thm1 : $v \in f[j+1\,..\,n]$
variant
  $n - j$
events
  ...
end

initialisation $\widehat{=}$
  status **ordinary**
  then
    act1 : $i := 1$
    act2 : $j := 0$
  end

search $\widehat{=}$
  status **ordinary**
  refines
    search
  when
    grd1 : $f(j+1) = v$
  with
    k : $j+1 = k$
  then
    act1 : $i := j+1$
  end

progress $\widehat{=}$
  status **convergent**
  when
    grd1 : $f(j+1) \neq v$
  then
    act1 : $j := j+1$
  end

- Among others, one numeric variant PO is generated:
  - **progress**/NAT

Proof Obligation progress/NAT

axm1
axm2
axm3
thm1 of ctx_0
inv1 (abstract)
inv1 (concrete)
inv2 (concrete)
thm1 of m_1a
grd1 (concrete)
$\vdash$
variant is a natural number

$n \in \mathbb{N}$
$f \in 1 \,..\, n \rightarrow D$
$v \in \mathrm{ran}(f)$
$n \in \mathbb{N}1$
$i \in 1 \,..\, n$
$j \in 0 \,..\, n-1$
$v \notin f[1 \,..\, j]$
$v \in f[j+1 \,..\, n]$
$f(j+1) \neq v$
$\vdash$
$n - j \in \mathbb{N}$

machine
  m_1a
refines
  m_0a
...
variant
  $n - j$
events
  ...
end

progress $\widehat{=}$
  status
    convergent
  when
    grd1 : $f(j+1) \neq v$
  then
    act1 : $j := j + 1$
  end

Purpose of the Set Variant PO (FIN)

- Ensuring that a proposed set variant is indeed a finite set
- The name of this PO is:

FIN

Formal Definition of the Set Variant (FIN)

machine
  m
refines
  ...
sees
  ...
variables
  v
invariants
  $J(s, c, v, w)$
events
  ...
variant
  $t(s, c, v)$
end

s	:	seen sets
c	:	seen constants
v	:	variables
$A(s, c)$	:	seen axioms
$I(s, c, v)$	:	abs. invts.
$J(s, c, v, w)$	:	conc. invts.
$t(s, c, v)$	:	set variant

Axioms Abstract invariants Concrete invariants $\vdash$ Finiteness of set variant	FIN	$A(s, c)$ $I(s, c, v)$ $J(s, c, v, w)$ $\vdash$ $\mathrm{finite}(t(s, c, v))$

Example in Mch m_1b Refining Mch m_0b (FIN)

machine
  m_1b
refines
  m_0b
sees
  ctx_0
variables
  i
  j
invariants
  inv1 : $j \in 0 \,..\, n-1$
  inv2 : $v \notin f[i \,..\, j]$
  thm1 : $v \in f[j+1 \,..\, n]$
variant
  $j \,..\, n$
events
  ...
end

initialisation $\widehat{=}$
  status ordinary
  then
    act1 : $i := 1$
    act2 : $j := 0$
  end

search $\widehat{=}$
  status ordinary
  refines search
  when
    grd1 : $f(j+1) = v$
  then
    act1 : $i := j + 1$
  end

progress $\widehat{=}$
  status convergent
  when
    grd1 : $f(j+1) \neq v$
  then
    act1 : $j := j + 1$
  end

- Among others, one finiteness PO is generated

Proof Obligation FIN

axm1 **axm2** **axm3** **thm1** of **ctx_0** **inv1** (abstract) **inv1** (concrete) **inv2** (concrete) **thm1** of **m_1a** $\vdash$ variant is finite	$n \in \mathbb{N}$ $f \in 1 \,..\, n \rightarrow D$ $v \in \mathrm{ran}(f)$ $n \in \mathbb{N}1$ $i \in 1 \,..\, n$ $j \in 0 \,..\, n-1$ $v \notin f[1 \,..\, j]$ $v \notin f[j+1 \,..\, n]$ $\vdash$ $\mathrm{finite}(j \,..\, n)$

machine
  m_1b
refines
  m_0b
...
variant
  $j \,..\, n$
events
  ...
end

Purpose of the Numeric Variant Decreasing PO (VAR)

- Ensuring that each convergent event decreases the proposed numeric variant
- For a convergent event "evt", the name of this PO is:
  evt/VAR

Numeric Variant Decreasing (VAR)

evt
  status
    convergent
  any x **where**
    $G(x,s,c,w)$
  then
    $v :| \; BAP(x,s,c,w,w')$
  end

s	:	seen sets
c	:	seen constants
v	:	variables
$A(s,c)$	:	seen axioms
$I(s,c,v)$	:	abs. invts.
$J(s,c,v,w)$	:	conc. invts.
evt	:	specific event
x	:	event parameters
$G(x,s,c,v)$	:	event guards
$BAP(x,s,c,w,w')$	:	event before-after predicate
$n(s,c,w)$	:	numeric variant

Axioms and theorems Abstract invariants and theorems Concrete invariants and theorems Guards of the event Before-after predicate of the event $\vdash$ Modified variant smaller than variant	evt/VAR	$A(s,c)$ $I(s,c,v)$ $J(s,c,v,w)$ $G(x,s,c,w)$ $BAP(x,s,c,w,w')$ $\vdash$ $n(s,c,w') < n(s,c,w)$

Example in Mch m_1a Refining Mch m_0a (VAR)

machine
  m_1a
refines
  m_0a
sees
  ctx_0
variables
  i
  j
invariants
  inv1 : $j \in 0 \,..\, n-1$
  inv2 : $v \notin f[1 \,..\, j]$
  thm1 : $v \in f[j+1 \,..\, n]$
variant
  $n - j$
events
  ...
end

initialisation $\widehat{=}$
  status ordinary
  then
    act1 : $i := 1$
    act2 : $j := 0$
  end

search $\widehat{=}$
  status ordinary
  refines search
  when
    grd1 : $f(j+1) = v$
  with
    k : $j+1 = k$
  then
    act1 : $i := j+1$
  end

progress $\widehat{=}$
  status convergent
  when
    grd1 : $f(j+1) \neq v$
  then
    act1 : $j := j+1$
  end

- Among others, one numeric variant decreasing PO is generated:
  - **progress**/VAR

Proof Obligation progress/VAR

axm1 **axm2** **axm3** **thm1** of **ctx_0** **inv1** (abstract) **inv1** (concrete) **inv2** (concrete) **thm1** of **m_1a** **grd1** (concrete) $\vdash$ variant is a natural number	$n \in \mathbb{N}$ $f \in 1\,..\,n \rightarrow D$ $v \in \mathrm{ran}(f)$ $n \in \mathbb{N}1$ $i \in 1\,..\,n$ $j \in 0\,..\,n-1$ $v \notin f[1\,..\,j]$ $v \in f[j+1\,..\,n]$ $f(j+1) = v$ $\vdash$ $n - (j+1) < n - j$

machine
  m_1a
refines
  m_0a
. . .
variant
  $n - j$
events
  . . .
end

progress $\widehat{=}$
  status
    convergent
  when
    grd1 : $f(j+1) \neq v$
  then
    act1 : $j := j+1$
  end

Purpose of the Set Variant Decreasing PO (VAR)

- Ensuring that each convergent event decreases the proposed set variant
- For a convergent event "evt", the name of this PO is:
  evt/VAR

Formal Def. of the Set Variant Decreasing PO (VAR)

evt
  status
    convergent
  any x **where**
    $G(x,s,c,w)$
  then
    $v :\mid BAP(x,s,c,w,w')$
  end

s	:	seen sets
c	:	seen constants
v	:	variables
$A(s,c)$	:	seen axioms
$I(s,c,v)$	:	abs. invts.
$J(s,c,v,w)$	:	conc. invts.
evt	:	specific event
x	:	event parameters
$G(x,s,c,v)$	:	event guards
$BAP(x,s,c,w,w')$	:	event before-after predicate
$t(s,c,w)$	:	set variant

Axioms and theorems Abstract invariants andtheorems Concrete invariants andtheorems Guards of the event Before-after predicate of the event $\vdash$ Modified variant strictly included in variant	evt/VAR	$A(s,c)$ $I(s,c,v)$ $J(s,c,v,w)$ $G(x,s,c,v)$ $BAP(x,s,c,w,w')$ $\vdash$ $t(s,c,w') \subset t(s,c,w)$

Example in Mch m_1b Refining Mch m_0b (VAR)

machine
  m_1b
refines
  m_0b
sees
  ctx_0
variables
  i
  j
invariants
  inv1 : $j \in 0\,..\,n-1$
  inv2 : $v \notin f[1\,..\,j]$
  thm1 : $v \in f[j+1\,..\,n]$
variant
  $j\,..\,n$
events
  . . .
end

initialisation $\widehat{=}$
  status ordinary
  then
    act1 : $i := 1$
    act2 : $j := 0$
  end

search $\widehat{=}$
  status ordinary
  refines search
  when
    grd1 : $f(j+1) = v$
  then
    act1 : $i := j+1$
  end

progress $\widehat{=}$
  status convergent
  when
    grd1 : $f(j+1) \neq v$
  then
    act1 : $j := j+1$
  end

- Among others, one variant decreasing PO is generated:
  - **progress**/VAR

Proof Obligation progress/VAR

axm1
axm2
axm3
thm1 of **ctx_0**
inv1 (abstract)
inv1 (concrete)
inv2 (concrete)
thm1 of **m_1a**
inv2 (concrete)
grd1 (concrete)
$\vdash$
variant is a natural number

$n \in \mathbb{N}$
$f \in 1\,..\,n \rightarrow D$
$v \in \mathrm{ran}(f)$
$n \in \mathbb{N}1$
$i \in 1\,..\,n$
$j \in 0\,..\,n-1$
$v \notin f[1\,..\,j]$
$v \in f[j+1\,..\,n]$
$f(j+1) = v$
$\vdash$
$j+1\,..\,n \subset j\,..\,n$

machine
  m_1b
refines
  m_0b
...
variant
  $j\,..\,n$
events
  ...
end

progress $\widehat{=}$
  status
    convergent
  when
    grd1 : $f(j+1) \neq v$
  then
    act1 : $j := j+1$
  end

Purpose of the Witness Feasibility PO (WFIS)

- Ensuring that each witness proposed in the witness predicate of a concrete event indeed exists
- For a concrete event "evt", and an abstract parameter x the name of this PO is:

evt/x/WFIS

Formal Definition of the Witness Feasibility PO (WFIS)

evt
  refines
    evt0
  any
    y
  where
    $H(y,s,c,w)$
  with
    $x : W(x,y,s,c,w)$
  then
    ...
  end

s	:	seen sets
c	:	seen constants
v	:	abstract variables
w	:	concrete variables
$A(s,c)$	:	seen axioms
$I(s,c,v)$	:	abs. invts.
$J(s,c,v,w)$	:	conc. invts.
evt	:	specific concrete event
x	:	abstract event parameter
y	:	concrete event parameter
$H(y,s,c,w)$	:	concrete event guards
$W(x,y,s,c,w)$	:	witness predicate

Axioms Abstract invariants Concrete invariants Concrete event guards $\vdash$ $\exists x \cdot$ Witness	$evt/x/$WFIS

$A(s,c)$
$I(s,c,v)$
$J(s,c,v,w)$
$H(y,s,c,w)$
$\vdash$
$\exists x \cdot W(x,y,s,c,w)$

Example in Mch m_1a Refining Mch m_0a (WFIS)

machine
  m_1a
refines
  m_0a
sees
  ctx_0
variables
  i
  j
invariants
  inv1 : $j \in 0\,..\,n-1$
  inv2 : $v \notin f[1\,..\,j]$
  thm1 : $v \in f[j+1\,..\,n]$
variant
  $n - j$
events
  ...
end

initialisation $\widehat{=}$
  status **ordinary**
  then
    act1 : $i := 1$
    act2 : $j := 0$
  end

search $\widehat{=}$
  status **ordinary**
  refines **search**
  when
    grd1 : $f(j+1) = v$
  with
    k : $j+1 = k$
  then
    act1 : $i := j+1$
  end

progress $\widehat{=}$
  status **convergent**
  when
    grd1 : $f(j+1) \neq v$
  then
    act1 : $j := j+1$
  end

- Among others, one witness feasibility PO is generated:
  - **search**/k/WFIS

Proof Obligation search/k/WFIS

axm1 **axm2** **axm3** **thm1** of **ctx_0** **inv1** (abstract) **inv1** (concrete) **inv2** (concrete) **thm1** of **m_1a** **grd1** (concrete) $\vdash$ $\exists k \cdot$ variant predicate	$n \in \mathbb{N}$ $f \in 1\,..\,n \to D$ $v \in \mathrm{ran}(f)$ $n \in \mathbb{N}1$ $i \in 1\,..\,n$ $j \in 0\,..\,n-1$ $v \notin f[1\,..\,j]$ $v \in f[j+1\,..\,n]$ $f(j+1) = v$ $\vdash$ $\exists k \cdot j+1 = k$

search $\widehat{=}$
  status **ordinary**
  refines **search**
  when
    grd1 : $f(j+1) = v$
  with
    k : $j+1 = k$
  then
    act1 : $i := j+1$
  end

Purpose of a Context Theorem PO (THM)

- Ensuring that a proposed context theorem is indeed provable
- Theorems are important in that they might simplify some proofs
- For a theorem "thm" in a context, the name of this PO is:

  thm/THM

Formal Definition of the Context Theorem PO (THM)

context
  ctx
extends
  ...
sets
  s
constants
  c
axioms
  $A(s,c)$
  ...
  thm : $P(s,c)$
  ...
end

s	:	seen sets
c	:	seen constants
$A(s,c)$	:	seen axioms and previous theorems
$P(s,c)$	:	specific theorem

Axioms $\vdash$ Theorem	thm/THM

$$\begin{array}{l} A(s,c) \\ \vdash \\ P(s,c) \end{array}$$

Example in Context ctx_0 (THM)

context
  ctx_0
sets
  D
constants
  n
  f
  v
axioms
  axm1 : $n \in \mathbb{N}$
  axm2 : $f \in 1..n \to D$
  axm3 : $v \in \mathrm{ran}(f)$
  thm1 : $n \in \mathbb{N}1$
end

- One theorem PO is generated: **thm1**/THM

Proof Obligation thm1/THM

axm1 **axm2** **axm3** $\vdash$ **thm1**

$n \in \mathbb{N}$ $f \in 1\,..\,n \rightarrow D$ $v \in \mathrm{ran}(f)$ $\vdash$ $n \in \mathbb{N}1$

Purpose of a Machine Theorem PO (THM)

- Ensuring that a proposed machine theorem is indeed provable
- Theorems are important in that they might simplify some proofs
- For a theorem "thm" in a machine, the name of this PO is:
  thm/THM

Formal Definition of the Machine Theorem PO (THM)

```
machine
  m0
refines
  ...
sees
  ...
variables
  v
invariants
  I(s, c, v)
  ...
  thm :  P(s, c, v)
  ...
events
  ...
end
```

s	:	seen sets
c	:	seen constants
v	:	variables
$A(s,c)$	:	seen axioms
$I(s,c,v)$	:	invariants and previous thms.
$P(s,c,v)$	:	specific theorem

Axioms Invariants $\vdash$ Theorem	*thm*/THM

$$\begin{array}{l} A(s,c) \\ I(s,c,v) \\ \vdash \\ P(s,c,v) \end{array}$$

Example in Mch m_1a Refining Mch m_0a (THM)

machine
  m_1a
refines
  m_0a
sees
  ctx_0
variables
  i
  j
invariants
  inv1 : $j \in 0\,..\,n-1$
  inv2 : $v \notin f[1\,..\,j]$
  thm1 : $v \in f[j+1\,..\,n]$
variant
  $n-j$
events
  ...
end

- Among others, one theorem PO is generated: **thm1**/THM

Proof Obligation thm1/THM

axm1 **axm2** **axm3** **thm1** of **ctx_0** **inv1** (abstract) **inv1** (concrete) **inv2** (concrete) $\vdash$ **thm1** of **m_1a**	$n \in \mathbb{N}$ $f \in 1\,..\,n \to D$ $v \in \mathrm{ran}(f)$ $n \in \mathbb{N}1$ $i \in 1\,..\,n$ $j \in 0\,..\,n-1$ $v \notin f[1\,..\,j]$ $\vdash$ $v \in f[j+1\,..\,n]$

Purpose of a Well-definedness PO (WD)

- Ensuring that a potentially ill-defined axiom, theorem, invariant, guard, action, variant, or witness is indeed well-defined
- For a given modeling element (axm, thm, inv, grd, act), or a variant, or a witness x in an event evt, the names are:

  axm/WD, thm/WD, inv/WD, grd/WD, act/WD, VWD, evt/x/WWD

Formal Definition of the Well-definedness PO (WD)

- It depends on the potentially ill-defined expression

$\mathrm{inter}(S)$	$S \neq \varnothing$
$\bigcap x \cdot x \in S \wedge P(x) \mid T(x)$	$\exists x \cdot x \in S \wedge P(x)$
$f(E)$	f is a partial function $E \in \mathrm{dom}(f)$
E / F	$F \neq 0$
E mod F	$F \neq 0$
$\mathrm{card}(S)$	$\mathrm{finite}(S)$
$\min(S)$	$S \subseteq \mathbb{Z}$ $\exists x \cdot x \in \mathbb{Z} \wedge (\forall n \cdot n \in S \Rightarrow x \leq n)$
$\max(S)$	$S \subseteq \mathbb{Z}$ $\exists x \cdot x \in \mathbb{Z} \wedge (\forall n \cdot n \in S \Rightarrow x \geq n)$

Examples in Machine m_0a (WD)

```
context
   ctx_0
sets   D
constants   n, f, v
axioms
   axm1 : n ∈ ℕ
   axm2 : f ∈ 1..n → D
   axm3 : v ∈ ran(f)
   thm1 : n ∈ ℕ1
end
```

```
machine
   m_0a
sees   ctx_0
variables
   i
invariants
   inv1 :  i ∈ 1 .. n
events
   . . .
end
```

```
initialisation  ≙
   status
      ordinary
   then
      act1 :  i := 1
   end
```

```
search  ≙
   status
      ordinary
   any
      k
   where
      grd1 : k ∈ 1 .. n
      grd2 : f(k) = v
   then
      act1 :  i := k
   end
```

- One well-definedness PO is generated:

  - search/grd2/WD

Proof Obligation search/grd2/WD

axm1 **axm2** **axm3** **thm1** **inv1** **grd1** $\vdash$ WD conditions for **grd2**	$n \in \mathbb{N}$ $f \in 1\,..\,n \rightarrow D$ $v \in \mathrm{ran}(f)$ $n \in \mathbb{N}1$ $i \in 1\,..\,n$ $k \in 1\,..\,n$ $\vdash$ $k \in \mathrm{dom}(f) \wedge f \in \mathbb{Z} \nrightarrow D$

Grd Strengthening when Merging Abs Events (MRG)

```
evt01
  any
    x
  where
    G1(x, s, c, v)
  then
    A
  end
```

```
evt02
  any
    x
  where
    G2(x, s, c, v)
  then
    A
  end
```

```
evt
  refines
    evt01
    evt02
  any
    x
  where
    H(x, s, c, v)
  then
    A
  end
```

Axioms and theorems
Abstract invariants and theorems

Summary of all POs of the Examples (1)

- Context **ctx_0**
  - **thm1**/THM
- Machine **m_0a**
  - **initialisation/inv1**/INV
  - **search/gdr2**/WD
  - **search/inv1**/INV
- Machine **m_0b**
  - **initialisation/inv1**/INV
  - **search/inv1**/INV
  - **search/act1**/WD
  - **search/act1**/FIS

Summary of all POs of the Examples (2)

- Machine **m_1a**
  - **thm1**/THM
  - **initialisation/inv1**/INV
  - **initialisation/inv2**/INV
  - **search/gdr1**/WD
  - **search**/*k*/WFIS
  - **search/gdr1**/GRD
  - **search/gdr2**/GRD
  - **search/act1**/SIM
  - **progress/gdr1**/WD
  - **progress/inv1**/INV
  - **progress/inv2**/INV
  - **progress**/VAR
  - **progress**/NAT

Summary of all POs of the Examples (3)

- Machine **m_1b**
  - **thm1**/THM
  - FIN
  - **initialisation/inv1**/INV
  - **initialisation/inv2**/INV
  - **search/gdr1**/WD
  - **search/act1**/SIM
  - **progress/gdr1**/WD
  - **progress/inv1**/INV
  - **progress/inv2**/INV
  - **progress**/VAR