Summary of Event-B Proof Obligations

Jean-Raymond Abrial (ETHZ)

March 2008

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 static checks contexts or machine texts

- It decides then what is to be proved

- The outcome are various sequents, which are transmitted to the provers performing automatic or interactive proofs

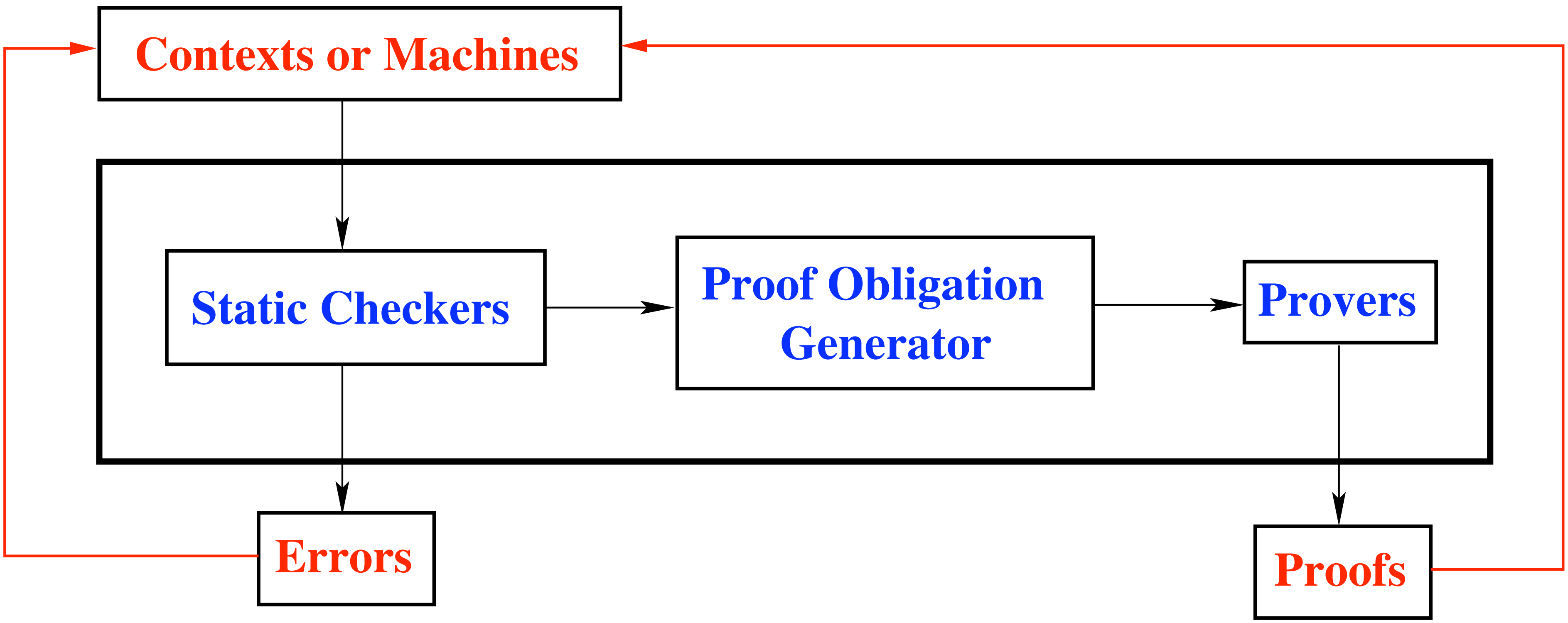
Summary of the Main Rodin Platform Kernel Tools

- 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 (INV slide 8)

- Non-deterministic action feasibility (FIS slide 13)

- Guard strengthening in a refinement (GRD slide 17)

- Simulation (SIM slide 21)

- Numeric variant (NAT slide 25)

- Set variant (FIN slide 29)

Various Kinds of Proof Obligations (cont'd)

- Variant decreasing (VAR slide 33)

- Feasibility of a non-deterministic witness (WFIS slide 41)

- Proving theorems (THM slide 45)

- Well-definedness (WD slide 53)

- Guard strengthening when merging abstract events (MRG slide 57)

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)

- 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)

```
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 and thms
$I(s,c,v)$	:	invariants and thms.
evt	:	specific event
x	:	event parameters
$G(x,s,c,v)$	:	event guards
$BAP(x,s,c,v,v')$	:	event before-after predicate
$inv(s,c,v')$	:	modified specific invariant

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

$$
\begin{array}{l}
A(s,c) \\
I(s,c,v) \\
G(x,s,c,v) \\
BAP(x,s,c,v,v') \\
\vdash \\
inv(s,c,v')
\end{array}
$$

- 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 : $n \in \mathbb{N}$
  axm2 : $f \in 1..n \rightarrow D$
  axm3 : $v \in \mathrm{ran}(f)$
theorems
  axm1 : $n \in \mathbb{N}1$
end

machine
  m_0a
sees
  ctx_0
variables
  i
invariants and thms.
  inv1 : $i \in 1\,..\,n$
events
  $\ldots$
end

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

search $\widehat{=}$
  status
    ordinary
  any
    k
  where
    grd1 : $k \in 1\,..\,n$
    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 $\widehat{=}$
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**

$$
\begin{array}{l}
n \in \mathbb{N} \\
f \in 1 \,..\, n \rightarrow D \\
v \in \operatorname{ran}(f) \\
n \in \mathbb{N}1 \\
i \in 1 \,..\, n \\
k \in 1 \,..\, n \\
\hline
f(k) = v \\
i' = k \\
\vdash \\
i' \in 1 \,..\, n
\end{array}
$$

$$
\begin{array}{l}
n \in \mathbb{N} \\
f \in 1 \,..\, n \rightarrow D \\
v \in \operatorname{ran}(f) \\
n \in \mathbb{N}1 \\
i \in 1 \,..\, n \\
k \in 1 \,..\, n \\
\hline
f(k) = v \\
\vdash \\
k \in 1 \,..\, n
\end{array}
$$

Simplification performed
by the PO Generator

search $\widehat{=}$
  status
    ordinary
  any
    k
  where
    grd1 : $k \in 1 \,..\, n$
    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 and thms
$I(s,c,v)$	:	invariants and thms.
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

$$
\begin{array}{l}
A(s,c) \\
I(s,c,v) \\
G(x,s,c,v) \\
\vdash \\
\exists v' \cdot BAP(x,s,c,v,v')
\end{array}
$$

Example in Machine m_0b (FIS)

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

machine
  m_0b
sees
  ctx_0
variables
  i
invariants and thms.
  inv1 : $i \in 1\,..\,n$
events
  $\ldots$
end

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

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

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

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

Formal Definition of the Guard Strengthening PO (GRD)

evt0	evt
evt0 **any** x **where** $g(x,s,c,v)$ $\ldots$ **then** $\ldots$ **end**	evt **refines** evt0 **any** y **where** $H(y,s,c,w)$ **with** $x : W(x,y,s,c,w)$ **then** $\ldots$ **end**

s	:	seen sets
c	:	seen constants
v	:	abstract variables
w	:	concrete variables
$A(s,c)$	:	seen axioms and thms
$I(s,c,v)$	:	abs. invts. and thms.
$J(s,c,v,w)$	:	conc. invts. and thms.
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 and thms. Concrete invariants and thms. 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 Machine m_1a Refining Machine m_0a (GRD)

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

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

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

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

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)

$$
\begin{array}{l}
n \in \mathbb{N} \\
f \in 1 \mathinner{..} n \rightarrow D \\
v \in \mathrm{ran}(f) \\
n \in \mathbb{N}1 \\
i \in 1 \mathinner{..} n \\
j \in 0 \mathinner{..} n \\
v \notin f[1 \mathinner{..} j] \\
v \in f[j+1 \mathinner{..} n] \\
\underline{f(j+1) = v} \\
\underline{j+1 = k} \\
\vdash \\
f(k) = v
\end{array}
$$

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 : $k \in 1 \mathinner{..} n$
    grd2 : $f(k) = v$
  then
    act1 : $i := k$
  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 both concrete and abstract events, the name of this PO is:

evt/act/SIM

Formal Definition of the Simulation PO (SIM)

evt0
evt0 **any** x **where** $\dots$ **then** $v :\mid BA1(v, v', \dots)$ **end**

evt
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** $w :\mid BA2(w, w', \dots)$ **end**

s	:	seen sets
c	:	seen constants
v	:	abstract vrbls
w	:	concrete vrbls
$A(s, c)$	:	seen axioms and thms
$I(s, c, v)$	:	abs. invts. and thms.
$J(s, c, v, w)$	:	conc. invts. and thms.
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 and thms. Concrete invariants and thms. Concrete event guards witness predicate witness predicate Concrete before-after predicate $\vdash$ Abstract before-after predicate	$evt/act/$SIM

$$
\begin{array}{l}
A(s, c) \\
I(s, c, v) \\
J(s, c, v, w) \\
H(y, s, c, w) \\
W1(x, y, s, c, w) \\
W2(y, v', s, c, w) \\
BA2(w, w', \dots) \\
\vdash \\
BA1(v, v', \dots)
\end{array}
$$

Example in Machine m_1a Refining Machine m_0a (SIM)

machine
  m_1a
refines
  m_0a
sees
  ctx_0
variables
  i
  j
invariants and thms.
  inv1 : $j \in 0 \,..\, n$
  inv2 : $v \notin f[1 \,..\, j]$
theorems
  thm1 : $v \in f[j+1 \,..\, n]$
variant
  $n - j$
events
  $\ldots$
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
axm2
axm3
thm1 of **ctx_0**
inv1 (abstract)
inv1 (concrete)
inv2 (concrete)
thm1 of **m_1a**
grd1 (concrete)
witness predicate
$\vdash$
before-after predicate (abstract)

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

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** $\ldots$ **sees** $\ldots$ **variables** v **invariants and thms.** $I(s,c,v)$ **theorems** $\ldots$ **events** $\ldots$ **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 and thms
$I(s,c,v)$	:	abs. invts. and thms.
$J(s,c,v,w)$	:	conc. invts. and thms.
evt	:	specific event
x	:	event parameters
$G(x,s,c,v)$	:	event guards
$n(s,c,v)$	:	numeric variant

Axioms Abstract invariants and thms. Concrete invariants and thms. Event guards $\vdash$ a numeric variant is a natural number	evt/NAT

$$
\begin{array}{l}
A(s,c) \\
I(s,c,v) \\
J(s,c,v,w) \\
G(x,s,c,v) \\
\vdash \\
n(s,c,v) \in \mathbb{N}
\end{array}
$$

Example in Machine m_1a Refining Machine m_0a (NAT)

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

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

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

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

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

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
  $\ldots$
sees
  $\ldots$
variables
  v
invariants and thms.
  $J(s,c,v,w)$
theorems
  $\ldots$
events
  $\ldots$
variant
  $t(s,c,v)$
end

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

Axioms Abstract invariants and thms. Concrete invariants and thms. $\vdash$ Finiteness of set variant	FIN

$$
\begin{array}{l}
A(s,c) \\
I(s,c,v) \\
J(s,c,v,w) \\
\vdash \\
\text{finite}(t(s,c,v))
\end{array}
$$

Example in Machine m_1b Refining Machine m_0b (FIN)

machine
  m_1b
refines
  m_0b
sees
  ctx_0
variables
  i
  j
invariants and thms.
  inv1 : $j \in 0\,..\,n$
  inv2 : $v \notin f[i\,..\,j]$
theorems
  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

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

```
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 and thms
$I(s, c, v)$	:	abs. invts. and thms.
$J(s, c, v, w)$	:	conc. invts. and thms.
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 Abstract invariants and thms. Concrete invariants and thms. Guards of the event Before-after predicate of the event ⊢ Modified variant smaller than variant	evt/VAR

$$\begin{array}{l} A(s, c) \\ I(s, c, v) \\ J(s, c, v, w) \\ G(x, s, c, w) \\ BAP(x, s, c, w, w') \\ \vdash \\ n(s, c, w') < n(s, c, w) \end{array}$$

Example in Machine m_1a Refining Machine m_0a (VAR)


```
machine
   m_1a
refines
   m_0a
sees
   ctx_0
variables
   i
   j
invariants and thms.
   inv1 :  j ∈ 0 .. n
   inv2 :  v ∉ f[1 .. j]
theorems
   thm1 :  v ∈ f[j + 1 .. n]
variant
   n − j
events
   ...
end
```

```
initialisation ≙
  status
    ordinary
  then
    act1 :  i := 1
    act2 :  j := 0
  end
```

```
search ≙
  status
    ordinary
  refines
    search
  when
    grd1 : f(j + 1) = v
  with
    k :  j + 1 = k
  then
    act1 :  i := j + 1
  end
```

```
progress ≙
  status
    convergent
  when
    grd1 : f(j + 1) ≠ v
  then
    act1 :  j := j + 1
  end
```

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

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

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 Definition of the Set Variant Decreasing PO (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 and thms
$I(s, c, v)$	:	abs. invts. and thms.
$J(s, c, v, w)$	:	conc. invts. and thms.
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 Abstract Invariants Concrete Invariants 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 Machine m_1b Refining Machine m_0b (VAR)

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

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

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

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

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

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 and thms
$I(s,c,v)$	:	abs. invts. and thms.
$J(s,c,v,w)$	:	conc. invts. and thms.
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 and thms. Concrete invariants and thms. Concrete event guards $\vdash$ $\exists x \cdot$ Witness	$evt/x/$WFIS

$$
\begin{array}{l}
A(s,c) \\
I(s,c,v) \\
J(s,c,v,w) \\
H(y,s,c,w) \\
\vdash \\
\exists x \cdot W(x,y,s,c,w)
\end{array}
$$

Example in Machine m_1a Refining Machine m_0a (WFIS)

machine
  m_1a
refines
  m_0a
sees
  ctx_0
variables
  i
  j
invariants and thms.
  inv1 : $j \in 0 \,..\, n$
  inv2 : $v \notin f[1 \,..\, j]$
theorems
  thm1 : $v \in f[j+1 \,..\, n]$
variant
  $n - j$
events
  $\ldots$
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	$n \in \mathbb{N}$ $f \in 1\,..\,n \rightarrow D$ $v \in \mathrm{ran}(f)$ $n \in \mathbb{N}1$ $i \in 1\,..\,n$ $j \in 0\,..\,n$ $v \notin f[1\,..\,j]$ $v \in f[j+1\,..\,n]$ $f(j+1) = v$ $\vdash$ $\exists k \cdot j + 1 = k$

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** $\ldots$ **sets** s **constants** c **axioms** $A(s,c)$ **theorems** $\ldots$ $thm: \quad P(s,c)$ $\ldots$ **end**

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

Axioms $\vdash$ Theorem	thm/THM

$$A(s,c) \vdash P(s,c)$$

Example in Context ctx_0 (THM)

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

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

Proof Obligation thm1/THM

axm1
axm2
axm3
$\vdash$
thm1

$$
\begin{array}{l}
\underline{n \;\in\; \mathbb{N}} \\
f \;\in\; 1\,..\,n \rightarrow D \\
\underline{v \in \operatorname{ran}(f)} \\
\vdash \\
n \;\in\; \mathbb{N}1
\end{array}
$$

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
$\dots$
sees
$\dots$
variables
v
invariants and thms.
$I(s,c,v)$
theorems
$\dots$
$thm: \quad P(s,c,v)$
$\dots$
events
$\dots$
end

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

Axioms Invariants ⊢ Theorem	thm/THM

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

Example in Machine m_1a Refining Machine m_0a (THM)

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

$$
\begin{array}{l}
n \;\in\; \mathbb{N} \\
f \;\in\; 1\,..\,n \rightarrow D \\
\underline{v \in \mathrm{ran}(f)} \\
n \;\in\; \mathbb{N}1 \\
i \in 1\,..\,n \\
j \in 0\,..\,n \\
\underline{v \notin f[1\,..\,j]} \\
\vdash \\
v \in f[j+1\,..\,n]
\end{array}
$$

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	$F \neq 0$
$E \bmod F$	$F \neq 0$
$\mathrm{card}(S)$	$\mathrm{finite}(S)$
$\min(S)$	$S \subseteq \mathbb{Z}$ $\exists x \cdot x \in \mathbb{Z}\ \wedge\ (\forall n \cdot n \in S\ \Rightarrow\ x \leq n)$
$\max(S)$	$S \subseteq \mathbb{Z}$ $\exists x \cdot x \in \mathbb{Z}\ \wedge\ (\forall n \cdot n \in S\ \Rightarrow\ x \geq n)$

Examples in Machine m_0a (WD)

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

machine
  m_0a
sees
  ctx_0
variables
  i
invariants and thms.
  inv1 : $i \in 1 \,..\, n$
events
  $\ldots$
end

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

search $\widehat{=}$
  status
    ordinary
  any
    k
  where
    grd1 : $k \in 1 \,..\, n$
    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**	$n \in \mathbb{N}$ $f \in 1..n \rightarrow D$ $v \in \mathrm{ran}(f)$ $n \in \mathbb{N}1$ $i \in 1..n$ $k \in 1..n$ $\vdash$ $k \in \mathrm{dom}(f) \wedge f \in \mathbb{Z} \mapsto D$

```
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
```

s	:	seen sets
c	:	seen constants
v	:	abstract vrbls
$A(s,c)$	:	seen axioms and thms
$I(s,c,v)$	:	abs. invts. and thms.
evt	:	concrete event
x	:	similar prm
$H(x,s,c,v)$	:	concrete guards
$G1(x,s,c,v)$	:	abstract event guards
$G2(x,s,c,v)$	:	abstract event guards
A	:	similar abs. and cnc. actions

Axioms Abstract invariants and thms. Concrete event guards $\vdash$ Disjunction of abstract guards	evt/MRG

$$
\begin{array}{l}
A(s,c) \\
I(s,c,v) \\
H(x,s,c,v) \\
\vdash \\
G1(x,s,c,v) \lor G2(x,s,c,v)
\end{array}
$$

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