HEINRICH HEINE
UNIVERSITÄT DÜSSELDORF

Meta-Predicates for Rodin

Sebastian Krings

Institut für Informatik
Heinrich-Heine-Universität Düsseldorf

Rodin Workshop 2016

Motivation

- Common Patterns in Event-B
  - PO for deadlock freedom
  - PO for deterministic behavior
- Realized by composing guards
  - in other guards
  - in invariants
- Can be automated!

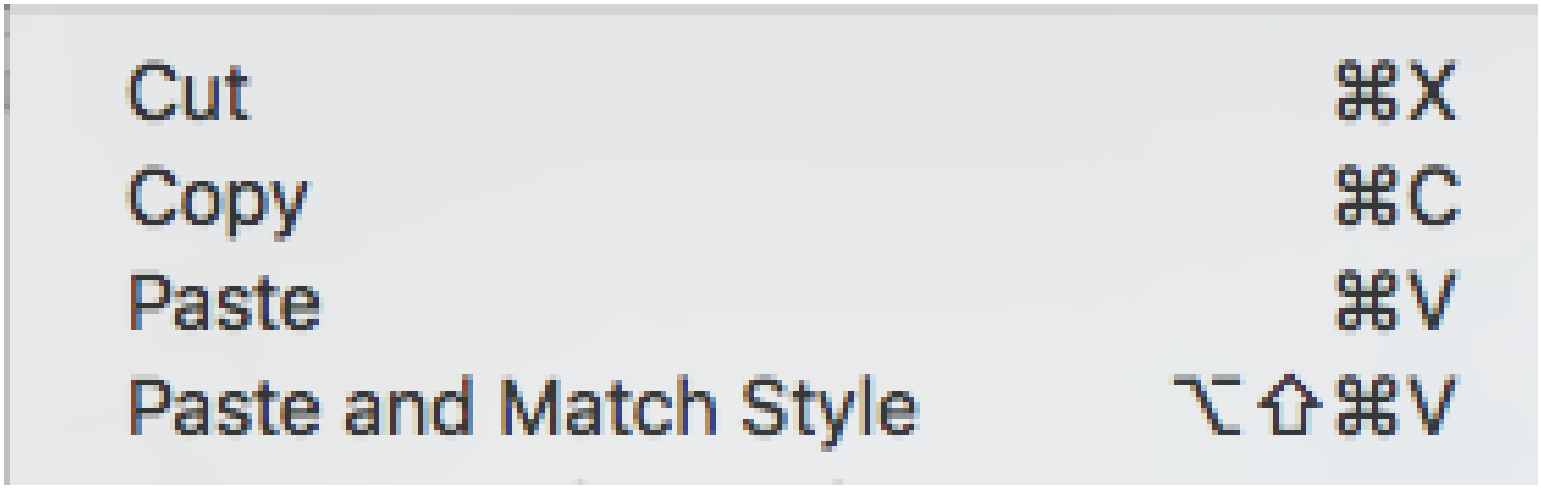
Deadlock Freedom - Example

```
event evt1
any
  x1
where
  x1 > 4
then
  counter := counter + x1
end

event evt2
any
  x2
where
  x2 < 2
then
  counter := counter − x2
end
```

$$\text{Deadlock Freedom} \Leftrightarrow \exists x1 \,.\, x1 > 4 \vee \exists x2 \,.\, x2 < 2$$

The Problem



But we know how to do it!

HEINRICH HEINE
UNIVERSITÄT DÜSSELDORF

We found copy and paste errors in the ABZ'14 case studies.

Meta-Predicate in Invariant


```
EXTENDED INVARIANTS
  ¬(enabled({error_case_event})) not theorem
INVARIANTS
  inv1:   pc ∈ N theorem
```

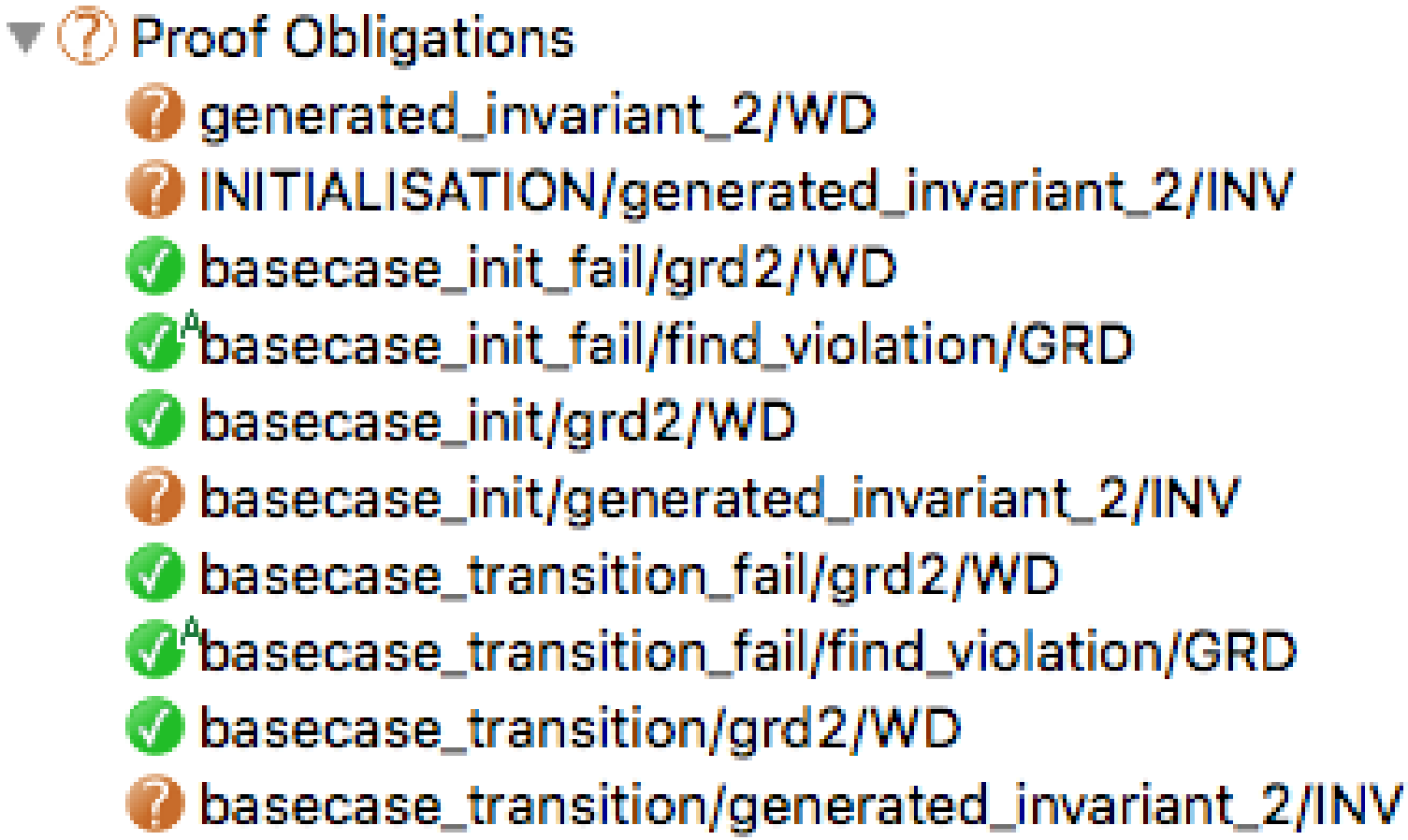
More Involved Invariant


```
EXTENDED INVARIANTS
o   enabled({recursive_case}) ⇒ ¬enabled({recursive_case_fail}) not theorem ›
```

Matching Guard

```
recursive_case_fail:     not extended ordinary ›
REFINES
  o    terminate_with_ce
ANY
  o   violating_state   ›
EXTENDED GUARDS
  o   ¬ enabled({recursive_case}) not theorem ›
WHERE
  o   grd1:   pc = recursive_case not theorem ›
  o   find_violation: PROPERTY(violating_state) = FALSE not theorem ›
THEN
  o   act1:   result = counterexample_found ›
  o   act2:   counterexample=violating_state ›
END
```

Result



Replaced by static checker, completely transparent to Rodin Platform

New Predicates

- *deadlock*, all events disabled

$$deadlock(Events) = \bigwedge_{e \in Events} \bigvee_{g \in guards(e)} \neg g$$

- *enabled*, all events enabled

$$enabled(Events) = \bigwedge_{e \in Events} \bigwedge_{g \in guards(e)} g$$

New Predicates Continued

- *controller*, exactly one enabled

$$controller(Events) = \bigvee_{e \in Events} (enabled(\{e\}) \wedge deadlock(Events \setminus \{e\}))$$

- *deterministic* order of execution

$$deterministic(Events) = controller(Events) \vee deadlock(Events)$$

Problematic Example

HEINRICH HEINE
UNIVERSITÄT DÜSSELDORF

```
event evt1
any
  x1
where
  x1 > 4
then
  counter := counter + x1
end

event evt2
any
  x2
where
  enabled(evt1)
  x2 < 2
then
  counter := counter − x2
end
```

Problematic Example

```
event evt1
any
  x1
where
  x1 > 4
then
  counter := counter + x1
end
```

```
event evt2
any
  x1, x2
where
  x1 > 4
  x2 < 2
then
  counter := counter-x2
end
```

```
event evt2
any
  x2
where
  #(x1).(x1 > 4)
  x2 < 2
then
  counter := counter-x2
end
```

Yet Another Problematic Example

HEINRICH HEINE
UNIVERSITÄT DÜSSELDORF

```
event evt1
any
  x1
where
  x1 > 4
then
  counter := counter + x1
end

event evt2
any
  x1
where
  enabled(evt1)
  x1 < 2
then
  counter := counter - x1
end
```

Problematic Example

```
event evt1
any
  x1
where
  x1 > 4
then
  counter := counter + x1
end
```

```
event evt2
any
  x1
where
  x1 > 4
  x1 < 2
then
  counter := counter-x1
end
```

```
event evt2
any
  x1
where
  #(x1).(x1 > 4)
  x1 < 2
then
  counter := counter-x1
end
```

Decide on Scoping

```
event evt1
any
  x1
where
  x1 > 4
then
  counter := counter + x1
end

event evt2
any
  x1
where
  ¬ enabled(evt1)
  x1 > 2
then
  counter := counter − x1
end
```

Decide on Scoping

```
event evt1
any
  x1
where
  x1 > 4
then
  counter := counter + x1
end
```

```
event evt2
any
  x1
where
  ¬x1 > 4
  x1 > 2
then
  counter := counter−x1
end
```

```
event evt2
any
  x1
where
  ¬∃x1.x1 > 4
  x1 > 2
then
  counter := counter−x1
end
```

Decide on Scoping

- Always introduce using quantifier
- Only introduce if new variable name
- Introduce using parameter
- Introduce preference

Conclusions

- Easy to implement using Rodin's API
- Less error-prone
  - No copy & paste
  - No missing updates
  - Expressiveness
- Useful for analysis:
  - Static checking
  - Constraint based verification
  - Model checking
  - Control flow
  - ...

Availability

HEINRICH HEINE
UNIVERSITÄT DÜSSELDORF

- Sources and Bugtracker
  `github.com/wysiib/RodinMetaPredicatesPlugin`
- Update Site for Rodin
  `www3.hhu.de/stups/rodin/meta_predicates/nightly`
- Currently only nightly builds

HEINRICH HEINE
UNIVERSITÄT DÜSSELDORF

Thank you!