2. Controlling Cars on a Bridge

Jean-Raymond Abrial

April 2008
- To present an example of system development

- Our approach: a series of more and more accurate models

- This approach is called refinement

- The models formalize the view of an external observer
- Each model will be analyzed and proved to be correct

- The aim is to obtain a system that will be correct by construction

- The correctness criterions correspond to a number of proof rules

- Proofs will be performed by using the sequent calculus

- Inference rules used in the sequent calculus will be reviewed
- The concepts of state and events for defining models
- A few principles of system development: essentially refinement
- A refresher on classical logic and simple arithmetic foundations
- A refresher on formal proofs
1. Presenting the requirement document (as in previous lecture)

2. Defining the strategy

3. Development of the initial model and the refinements

- Remark: During the development some theoretical background will be provided
- The system we are going to build is a piece of software connected to some equipment.

- There are two kinds of requirements:
  - those concerned with the equipment, labeled EQP,
  - those concerned with the function of the system, labeled FUN.

- The function of this system is to control cars on a narrow bridge.

- This bridge is supposed to link the mainland to a small island.
The system is controlling cars on a bridge between the mainland and an island

- This can be illustrated as follows
- The controller is equipped with two traffic lights with two colors.

| The system has two traffic lights with two colors: green and red | EQP-1 |
- One of the traffic lights is situated on the mainland and the other one on the island. Both are close to the bridge.

- This can be illustrated as follows
The traffic lights control the entrance to the bridge at both ends of it

- Drivers are supposed to obey the traffic light by not passing when a traffic light is red.

Cars are not supposed to pass on a red traffic light, only on a green one
- There are also some car sensors situated at both ends of the bridge.

- These sensors are supposed to detect the presence of cars intending to enter or leave the bridge.

- There are four such sensors. Two of them are situated on the bridge and the other two are situated on the mainland and on the island.

The system is equipped with four car sensors each with two states: on or off
The sensors are used to detect the presence of cars entering or leaving the bridge.

- The pieces of equipment can be illustrated as follows:
This system has two main constraints: the number of cars on the bridge and the island is limited and the bridge is one way.

<table>
<thead>
<tr>
<th>The number of cars on the bridge and the island is limited</th>
<th>FUN-2</th>
</tr>
</thead>
<tbody>
<tr>
<td>The bridge is one way or the other, not both at the same time</td>
<td>FUN-3</td>
</tr>
<tr>
<td>The system is controlling cars on a bridge between the mainland and an island</td>
<td>FUN-1</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>The number of cars on the bridge and the island is limited</td>
<td>FUN-2</td>
</tr>
<tr>
<td>The bridge is one way or the other, not both at the same time</td>
<td>FUN-3</td>
</tr>
<tr>
<td>The system has two traffic lights with two colors: green and red</td>
<td>EQP-1</td>
</tr>
<tr>
<td>---------------------------------------------------------------</td>
<td>-------</td>
</tr>
<tr>
<td>The traffic lights control the entrance to the bridge at both ends of it</td>
<td>EQP-2</td>
</tr>
<tr>
<td>Cars are not supposed to pass on a red traffic light, only on a green one</td>
<td>EQP-3</td>
</tr>
<tr>
<td>The system is equipped with four car sensors each with two states: on or off</td>
<td>EQP-4</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>The sensors are used to detect the presence of cars entering or leaving the bridge</td>
<td>EQP-5</td>
</tr>
</tbody>
</table>
- **Initial model**: Limiting the number of cars (FUN_2)

- **First refinement**: Introducing the one way bridge (FUN_3)

- **Second refinement**: Introducing the traffic lights (EQP_1,2,3)

- **Third refinement**: Introducing the sensors (EQP_4,5)
- It is very simple

- We do not consider at all the equipment: traffic lights and sensors

- We even do not consider the bridge

- We are just interested in the pair “island-bridge”

- We are focusing on FUN-2 (limited number of cars on island-bridge)
Two Events which can be Observed

ML_out

ML_in
- **STATIC PART** of the state: constant $d$ with axiom $\text{axm0.1}$

\[
\begin{array}{c}
\text{constant: } \ d \\
\text{axm0.1: } \ d \in \mathbb{N}
\end{array}
\]

- $d$ is the **maximum number of cars** allowed in the Island-Bridge

- $\text{axm0.1}$ states that $d$ is a **natural number**

- Constant $d$ is a member of the set $\mathbb{N} = \{0, 1, 2, \ldots\}$
- DYNAMIC PART: variable $v$ with invariants $\text{inv0}_1$ and $\text{inv0}_2$

<table>
<thead>
<tr>
<th>variable: $n$</th>
<th>inv0_1: $n \in \mathbb{N}$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>inv0_2: $n \leq d$</td>
</tr>
</tbody>
</table>

- $n$ is the effective number of cars in the Island-Bridge

- $n$ is a natural number ($\text{inv0}_1$)

- $n$ is always smaller than or equal to $d$ ($\text{inv0}_2$): this is $\text{FUN}_2$
- Labels `axm0_1`, `inv0_1`, etc. have been chosen systematically

- `axm` or `inv` recall the purpose: axioms of constants or invariants of variables

- The 0 as in `inv0_1` stands for the initial model.

- Later we shall have `inv1_1` for an invariant of refinement 1, etc.

- The 1 as in `inv0_1` is a serial number

- Any convention is valid as long as it is systematic
- This is the **first transition** (or event) that can be observed

- A car is leaving the mainland and entering the Island-Bridge

- The **number of cars** in the Island-Bridge has been **incremented**
- We can also observe a second transition (or event)

- A car leaving the Island-Bridge and re-entering the mainland

- The number of cars in the Island-Bridge has been decremented
- Event \texttt{ML\_out} \textbf{increments} the number of cars

\[
\text{\texttt{ML\_out}}
\begin{align*}
\quad n & := n + 1
\end{align*}
\]

- Event \texttt{ML\_in} \textbf{decrements} the number of cars

\[
\text{\texttt{ML\_in}}
\begin{align*}
\quad n & := n - 1
\end{align*}
\]

- An event is denoted by its \textbf{name} and its \textbf{action} (an assignment)
- These events are approximations for two reasons:

1. They might be refined (made more precise) later

2. They might be insufficient at this stage because not consistent with the invariant

- We shall have to perform a proof in order to check this consistency
Towards the Proof: Before-after Predicates

- To each event can be associated a **before-after predicate**

- It denotes the *relationship* holding between the *values* of the variable *just before* and *just after* the event occurrence

- The **before-value** is denoted by the **variable name**, say \( n \)

- The **after-value** is denoted by the **variable name** *primed*, say \( n' \)
The Events

\[
\text{ML\_out} \quad n := n + 1 \\
\text{ML\_in} \quad n := n - 1
\]

The corresponding before-after predicates

\[
\begin{align*}
n' &= n + 1 \\
n' &= n - 1
\end{align*}
\]

These representations are equivalent
- The before-after predicates we have shown are very simple

\[ n' = n + 1 \quad n' = n - 1 \]

- The after-value \( n' \) is defined as a function of the before-value \( n \)

- This is because the corresponding events are deterministic

- In further lectures, we shall consider some non-deterministic events:

\[ n' \in \{ n + 1, n + 2 \} \]
- When we wrote events `ML_out` and `ML_in` we did not take into accounts invariants `inv0_1` and `inv0_2`.

- There is no reason a priori for these invariants to be preserved.

- This has to be proved rigorously.

- We have to clarify what we have to prove.

- We are going to define some proof obligations also called verification conditions.
- Let us consider invariant $\text{inv0_1}$

\[ n \in \mathbb{N} \]

- And let us consider event $\text{ML\_out}$ with before-after predicate

\[ n' = n + 1 \]

- Preserving $\text{inv0_1}$ means that we have (just after $\text{ML\_out}$):

\[ n' \in \mathbb{N} \quad \text{that is} \quad n + 1 \in \mathbb{N} \]
- Under hypothesis $n \in \mathbb{N}$ then $n + 1 \in \mathbb{N}$ is provable

- This can be written as follows

\[ n \in \mathbb{N} \vdash n + 1 \in \mathbb{N} \]

- This statement is called a **sequent** (more later)

- This can be **generalized** (next slide)
- We collectively denote our set of constants by $c$

- We denote our set of axioms by $A(c): A_1(c), A_2(c), \ldots$

- We collectively denote our set of variables by $v$

- We denote our set of invariants by $I(c, v): I_1(c, v), I_2(c, v), \ldots$
- We are given an event with before-after predicate $v' = E(c, v)$

- We have to prove the following in order to preserve invariant $I_i(c, v)$:

$$A(c), I(c, v) \vdash I_i(c, E(c, v))$$

- It says: prove $I_i(c, E(c, v))$ under hypotheses $A(c)$ and $I(c, v)$

- We have given the name INV to this proof obligation
A(c), I(c, v) ⊢ I_i(c, E(c, v))  

- Just before occurrence of the event represented by \( v' = E(c, v) \), \( A(c) \) as well as \( I(c, v) \) clearly holds. We can then assume them

- Just after the occurrence, invariant \( I_i(c, v) \) has become \( I_i(c, v') \), that is \( I_i(c, E(c, v)) \)

- This statement \( I_i(c, E(c, v)) \) must hold since we claimed that \( I_i(c, v) \) was an invariant
- The proof obligation

\[
A(c), I(c, v) \vdash I_i(c, E(c, v))
\]

INV

can be re-written vertically as follows:

<table>
<thead>
<tr>
<th>Axioms</th>
<th>Invariants</th>
<th>( A(c) )</th>
<th>( I(c, v) )</th>
<th>( I_i(c, E(c, v)) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( A(c) )</td>
<td>( I(c, v) )</td>
<td>( A(c) )</td>
<td>( I(c, v) )</td>
<td>( I_i(c, E(c, v)) )</td>
</tr>
<tr>
<td>( I(c, v) )</td>
<td>( I(c, E(c, v)) )</td>
<td>( I_i(c, E(c, v)) )</td>
<td>( I_i(c, E(c, v)) )</td>
<td>( I_i(c, E(c, v)) )</td>
</tr>
</tbody>
</table>
- A **sequent** is a formal statement of the following shape

\[ \text{H} \vdash \text{G} \]

- The symbol "\(\vdash\)" is called the **turnstile** (it corresponds to verb "yield")

- **H** denotes a set of **predicates**: the **hypotheses** (or **assumptions**)

- **G** denotes a predicate: the **goal** (or **conclusion**)

- It can be read: "**Assumptions H yield conclusion G**"
- We have two events

\[
\text{ML\_out} \\
n := n + 1
\]

\[
\text{ML\_in} \\
n := n - 1
\]

- And two invariants

\[
\text{inv0\_1: } n \in \mathbb{N}
\]

\[
\text{inv0\_2: } n \leq d
\]

- Thus, four statements have to be proved
To be Proved for event ML\_out and invariant inv0\_1

\[
\begin{align*}
\text{ML\_out} \\
 n & := n + 1
\end{align*}
\]

\[
(n' = n + 1)
\]

Axiom axm0\_1
Invariant inv0\_1
Invariant inv0\_2
\[ \vdash \]
Modified Invariant inv0\_1

\[
\begin{align*}
d & \in \mathbb{N} \\
n & \in \mathbb{N} \\
n & \leq d \\
\vdash \\
n + 1 & \in \mathbb{N}
\end{align*}
\]

- This proof obligation is named: ML\_out / inv0\_1 / INV
To be Proved for event ML\_out and invariant inv0\_2

\[
\begin{align*}
\text{ML\_out} \\
n &:= n + 1
\end{align*}
\]

\[
(n' = n + 1)
\]

Axiom \texttt{axm0.1}  
Invariant \texttt{inv0.1}  
Invariant \texttt{inv0.2}  
\[
\vdash \text{Modified Invariant } \texttt{inv0.2}
\]

\[
d \in \mathbb{N} \\
n \in \mathbb{N} \\
n \leq d \\
\vdash n + 1 \leq d
\]

- This proof obligation is named: \texttt{ML\_out / inv0\_2 / INV}
To be Proved for event ML\_in and invariant inv0\_1

\[
ML\_in \\
\begin{align*}
n & := n - 1 \\
(n' & = n - 1)
\end{align*}
\]

Axiom \textbf{axm0\_1} \\
Invariant \textbf{inv0\_1} \\
Invariant \textbf{inv0\_2}

\[
\vdash \text{Modified Invariant} \textbf{inv0\_1}
\]

\[
\begin{align*}
d & \in \mathbb{N} \\
n & \in \mathbb{N} \\
n & \leq d \\
\vdash \\
n - 1 & \in \mathbb{N}
\end{align*}
\]

- This proof obligation is named: \textbf{ML\_in / inv0\_1 / INV}
To be Proved for event ML\_in and invariant inv0\_2

\begin{align*}
\text{ML\_in} & \\
n := n - 1 \quad & (n' = n - 1)
\end{align*}

Axiom axm0\_1

Invariant inv0\_1

Invariant inv0\_2

\vdash

Modified Invariant inv0\_2

\begin{align*}
\begin{array}{c}
d \in \mathbb{N} \\
n \in \mathbb{N} \\
n \leq d \quad & \vdash \\
n - 1 \leq d
\end{array}
\end{align*}

- This proof obligation is named: ML\_in / inv0\_2 / INV
Summary of What is to be Proved

**ML\_out / inv0\_1 / INV**

\[
\begin{align*}
    d & \in \mathbb{N} \\
    n & \in \mathbb{N} \\
    n & \leq d \\
    \vdash n + 1 & \in \mathbb{N}
\end{align*}
\]

**ML\_out / inv0\_2 / INV**

\[
\begin{align*}
    d & \in \mathbb{N} \\
    n & \in \mathbb{N} \\
    n & \leq d \\
    \vdash n + 1 & \leq d
\end{align*}
\]

**ML\_in / inv0\_1 / INV**

\[
\begin{align*}
    d & \in \mathbb{N} \\
    n & \in \mathbb{N} \\
    n & \leq d \\
    \vdash n - 1 & \in \mathbb{N}
\end{align*}
\]

**ML\_in / inv0\_2 / INV**

\[
\begin{align*}
    d & \in \mathbb{N} \\
    n & \in \mathbb{N} \\
    n & \leq d \\
    \vdash n - 1 & \leq d
\end{align*}
\]
In the first step, we remove some irrelevant hypotheses.

In the second and final step, we accept the sequent as it is.
- In the previous slide, we have applied some implicit rules:
  - For simplifying given sequents
  - For asserting that a sequent is proved without justifications

- In order to be very precise, we make these rules more explicit

- Such rules are called rules of inference
- The first rule of inference can be stated as follows:

\[
\frac{H_1 \vdash G}{H_1, H_2 \vdash G} \quad \text{MON}
\]

- Above the horizontal line, we have a sequent called the **antecedent** (this predicate can be **missing** or on the contrary made **multiple**)

- Below the horizontal line we have a sequent called the **consequent**

- To prove the consequent, **it is sufficient** to prove the antecedent
- The Second Peano Axiom

\[ n \in \mathbb{N} \vdash n + 1 \in \mathbb{N} \]

\[ 0 < n \vdash n - 1 \in \mathbb{N} \]

- Such rules are called **axioms** (because they have no antecedents)
- Such rules are given here without demonstration

\[
\text{INC}
\]
\[
\begin{array}{c}
\text{n} < \text{m} \vdash \text{n} + 1 \leq \text{m}
\end{array}
\]

\[
\text{DEC}
\]
\[
\begin{array}{c}
\text{n} \leq \text{m} \vdash \text{n} - 1 \leq \text{m}
\end{array}
\]
Meta-variables in Rules of Inference

- This rule denotes the 2nd Peano axiom in very general terms:

\[
\begin{array}{c}
\text{n} \in \mathbb{N} \quad \vdash \quad \text{n} + 1 \in \mathbb{N}
\end{array}
\]

\text{P2}

- It is a rule schema where \( \text{n} \) is called a meta-variable

- It can be applied to the following sequent by pattern matching:

\[
\begin{array}{c}
a + b \in \mathbb{N} \quad \vdash \quad a + b + 1 \in \mathbb{N}
\end{array}
\]
- A proof is (for the moment) just a sequence of sequents

\[
\begin{align*}
    d & \in \mathbb{N} \\
    n & \in \mathbb{N} \\
    n & \leq d \\
    \vdash n + 1 & \in \mathbb{N}
\end{align*}
\]

\[
\begin{align*}
    n & \in \mathbb{N} \\
    \vdash n + 1 & \in \mathbb{N}
\end{align*}
\]

- The sequents are separated by the rule (name) connecting them

- A last rule name (with no antecedent) terminates the sequence
- We put a ? to indicate that we have no rule to apply

- The proof fails: we cannot conclude with rule INC ($n < d$ needed)
- The proof fails: we cannot conclude with rule $\text{P2}'$ ($0 < n$ needed)
\( d \in \mathbb{N} \)
\( n \in \mathbb{N} \)
\( n \leq d \)

\[ \vdash n - 1 \leq d \]

**MON**

\( n \leq d \)
\[ \vdash n - 1 \leq d \]

**DEC**

\[ n \leq m \vdash n - 1 \leq m \]

**DEC**
- We needed hypothesis \( n < d \) to prove \( \text{ML\_out} / \text{inv0\_2} / \text{INV} \)

- We needed hypothesis \( 0 < n \) to prove \( \text{ML\_in} / \text{inv0\_1} / \text{INV} \)

\[
\begin{align*}
\text{ML\_out} & \\
n & := n + 1
\end{align*}
\]

\[
\begin{align*}
\text{ML\_in} & \\
n & := n - 1
\end{align*}
\]

- We are going to add \( n < d \) as a guard to event \( \text{ML\_out} \)

- We are going to add \( 0 < n \) as a guard to event \( \text{ML\_in} \)
Improving the Events: Introducing Guards

<table>
<thead>
<tr>
<th>ML_out</th>
<th>ML_in</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>when</strong></td>
<td><strong>when</strong></td>
</tr>
<tr>
<td>$n &lt; d$</td>
<td>$0 &lt; n$</td>
</tr>
<tr>
<td><strong>then</strong></td>
<td><strong>then</strong></td>
</tr>
<tr>
<td>$n := n + 1$</td>
<td>$n := n - 1$</td>
</tr>
<tr>
<td><strong>end</strong></td>
<td><strong>end</strong></td>
</tr>
</tbody>
</table>

- We are adding **guards** to the events

- The guard is the **necessary condition** for an event to “occur”
- Given \( c \) with axioms \( A(c) \) and \( v \) with invariants \( I(c, v) \)

- Given an event with guard \( G(c, v) \) and b-a predicate \( v' = E(c, v) \)

- We modify the Invariant Preservation Rule as follows:

\[
\begin{array}{c|c}
A(c) & \text{INV} \\
I(c, v) & \\
G(c, v) & \\
\vdash & \\
I_i(c, E(c, v)) & \\
\end{array}
\]
The Modified Rule

<table>
<thead>
<tr>
<th>Axiomss</th>
<th>Invariants</th>
</tr>
</thead>
<tbody>
<tr>
<td>Guard of the event</td>
<td>⊢ Modified Invariant</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>⊢</th>
<th>A(c)</th>
</tr>
</thead>
<tbody>
<tr>
<td>⊢</td>
<td>I(c, v)</td>
</tr>
<tr>
<td>⊢</td>
<td>G(c, v)</td>
</tr>
</tbody>
</table>

| INV | \[ I_i(c, E(c, v)) \] |
- Adding a new assumptions to a sequent does not modify a proof

\[
\frac{
  d \in \mathbb{N} \\
  n \in \mathbb{N} \\
  n \leq d \\
  n < d
}{n + 1 \in \mathbb{N}} \quad \text{MON}
\]

\[
\frac{n \in \mathbb{N}}{n + 1 \in \mathbb{N}} \quad \text{P2}
\]
\[
\begin{align*}
\text{d} & \in \mathbb{N} \\
n & \in \mathbb{N} \\
n & \leq d \\
n & < d \\
\vdash \\
n + 1 & \leq d
\end{align*}
\]

- We can conclude now with rule \textbf{INC}

\[
\begin{align*}
\text{n} & < \text{m} \\
\vdash \text{n} + 1 & \leq \text{m}
\end{align*}
\]

\text{MON} \\
\text{INC}
A Formal Proof of: ML\_in / inv0\_1 / INV

\[
\begin{align*}
  d & \in \mathbb{N} \\
  n & \in \mathbb{N} \\
  n & \leq d \\
  0 & < n \\
  \vdash \quad n - 1 & \in \mathbb{N}
\end{align*}
\]

- We can now conclude with rule P2'

\[
\begin{align*}
  0 & < n \\
  \vdash\quad n - 1 & \in \mathbb{N}
\end{align*}
\]

\[
\begin{align*}
  0 & < n \\
  \vdash\quad n - 1 & \in \mathbb{N}
\end{align*}
\]

\[\text{P2'}\]
\[
\begin{align*}
  d & \in \mathbb{N} \\
n & \in \mathbb{N} \\
n & \leq d \\
\vdash n - 1 & \leq d
\end{align*}
\]

**MON**

\[
\begin{align*}
n & \leq d \\
\vdash n - 1 & \leq d
\end{align*}
\]

**DEC**

\[
\begin{align*}
n & \leq m \\
\vdash n - 1 & \leq m
\end{align*}
\]

**DEC**
<table>
<thead>
<tr>
<th>Statement</th>
<th>Proof</th>
</tr>
</thead>
<tbody>
<tr>
<td>(d \in \mathbb{N})</td>
<td>(d \in \mathbb{N})</td>
</tr>
<tr>
<td>(n \in \mathbb{N})</td>
<td>(n \in \mathbb{N})</td>
</tr>
<tr>
<td>(n \leq d)</td>
<td>(n \leq d)</td>
</tr>
<tr>
<td>(n &lt; d)</td>
<td>(n &lt; d)</td>
</tr>
<tr>
<td>(\vdash n + 1 \in \mathbb{N})</td>
<td>(\vdash n + 1 \leq d)</td>
</tr>
<tr>
<td>(d \in \mathbb{N})</td>
<td>(d \in \mathbb{N})</td>
</tr>
<tr>
<td>(n \in \mathbb{N})</td>
<td>(n \in \mathbb{N})</td>
</tr>
<tr>
<td>(n \leq d)</td>
<td>(n \leq d)</td>
</tr>
<tr>
<td>(\color{red}0 &lt; n)</td>
<td>(\color{red}0 &lt; n)</td>
</tr>
<tr>
<td>(\vdash n - 1 \in \mathbb{N})</td>
<td>(\vdash n - 1 \leq d)</td>
</tr>
</tbody>
</table>
- Our system must be initialized (with no car in the island-bridge)

- The initialization event is never guarded

- It does not mention any variable in the right hand side of :=

- It before-after predicate is just an after predicate

\[
\begin{align*}
\text{init} \\
\quad n &:= 0 \\
\text{After predicate} \\
\quad n' &= 0
\end{align*}
\]
- Given $c$ with axioms $A(c)$ and $v$ with invariants $I(c, v)$

- Given an init event with after predicate $v' = K(c)$

- The Invariant Establishment Rule is the following:

\[
\begin{array}{c|c|c}
\text{Axioms} & A(c) & \text{INV} \\
\downarrow & \downarrow & \downarrow \\
\text{Modified Invariant} & I_i(c, K(c)) & \\
\end{array}
\]
Applying the Invariant Establishment Rule

\[
\text{axm0\_1} \\
\vdash \\
\text{Modified inv0\_1}
\]

\[
\begin{array}{c}
d \in \mathbb{N} \\
\vdash \\
0 \in \mathbb{N}
\end{array}
\]

\text{inv0\_1} / INV

\[
\text{axm0\_1} \\
\vdash \\
\text{Modified inv0\_2}
\]

\[
\begin{array}{c}
d \in \mathbb{N} \\
\vdash \\
0 \leq d
\end{array}
\]

\text{inv0\_2} / INV
- First Peano Axiom

\[ \vdash 0 \in \mathbb{N} \]

- Third Peano Axiom (slightly modified)

\[ n \in \mathbb{N} \vdash 0 \leq n \]
Proofs

\[
\begin{align*}
\text{MON} & \quad \vdash 0 \in \mathbb{N} \\
\text{P1} & \quad \vdash 0 \in \mathbb{N} \\
\text{P3} & \quad \vdash 0 \leq d \\
\end{align*}
\]
- It is possible for the system to be blocked if both guards are false

- We do not want this to happen

- We figure out that one important requirement was missing

Once started, the system should work for ever

FUN-4
- Given \( c \) with axioms \( A(c) \) and \( v \) with invariants \( I(c, v) \)

- Given the guards \( G_1(c, v), \ldots, G_m(c, v) \) of the events

- We have to prove the following:

\[
\begin{array}{c|c}
A(c) \\
I(c, v) \\
\hline
\vdash \\
G_1(c, v) \lor \ldots \lor G_m(c, v) & \text{DLF}
\end{array}
\]
To be Proved

\begin{align*}
\text{axm0\_1} \\
\text{inv0\_1} \\
\text{inv0\_2}
\end{align*}
\quad\quad
\begin{align*}
d \in \mathbb{N} \\
n \in \mathbb{N} \\
n \leq d
\end{align*}
\quad\quad
\begin{align*}
\vdash \\
\quad n < d \lor 0 < n
\end{align*}

- This cannot be proved \textit{with the inference rules we have so far}

- \( n \leq d \) can be replaced by \( n = d \lor n < d \)

- We have then to perform a \textit{proof by cases}:
  - when \( n = d \)
  - when \( n < d \)
- **Proof by cases.** Note that we have two antecedents

\[
\begin{align*}
H, P & \vdash R \\
\hline
H, Q & \vdash R \\
\hline
H, P \lor Q & \vdash R
\end{align*}
\]

**OR_L**

- **Choice for proving a disjunctive goal**

\[
\begin{align*}
H & \vdash P \\
\hline
H & \vdash P \lor Q
\end{align*}
\]

**OR_R1**

\[
\begin{align*}
H & \vdash Q \\
\hline
H & \vdash P \lor Q
\end{align*}
\]

**OR_R2**
\[ d \in \mathbb{N} \]
\[ n \in \mathbb{N} \]
\[ n \leq d \]
\[ \vdash n < d \lor 0 < n \]
\( n \leq d \downarrow \)
\( \vdash n < d \lor 0 < n \)

OR_L

\( n < d \)
\( \vdash n < d \lor 0 < n \)

\( n = d \)
\( \vdash n < d \lor 0 < n \)
- The first ? seems to be obvious

- The second ? can be (partially) solved by applying the equality
- The basic property of a sequent

\[
\begin{align*}
\text{HYP} & \quad \frac{P \vdash P}{\text{HYP}} \\
\end{align*}
\]

- Applying an equality (from left to right and vice-versa)

\[
\begin{align*}
\text{EQ_LR} & \quad \frac{H_F, E = F \vdash P_F}{H_E, E = F \vdash P_E} \\
\text{EQ_RL} & \quad \frac{H_E, E = F \vdash P_E}{H_F, E = F \vdash P_F}
\end{align*}
\]
- We still have a problem: \( d \) must be positive!
- If \( d \) is equal to 0, then \textit{no car can ever enter the Island-Bridge}

\[ \text{axm0\_2:} \quad 0 < d \]
- Thanks to the proofs, we discovered 3 errors

- They were corrected by:
  - adding guards to both events
  - adding an axiom
- We have seen three Proof Rules:

  - The **Invariant Establishment** Rule: INV
  - The **Invariant Preservation** Rule: INV
  - The **Deadlock Freeness** Rule (not mandatory): DLF
Proof Rules (cont’d)

<table>
<thead>
<tr>
<th>Axioms</th>
<th>INV</th>
</tr>
</thead>
<tbody>
<tr>
<td>⊢ Modified Invariant</td>
<td>INV</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Axioms</th>
<th>INV</th>
</tr>
</thead>
<tbody>
<tr>
<td>⊢ Guard of the event</td>
<td>INV</td>
</tr>
<tr>
<td>⊢ Modified Invariant</td>
<td>INV</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Axioms</th>
<th>DLF</th>
</tr>
</thead>
<tbody>
<tr>
<td>⊢ Invariants</td>
<td>DLF</td>
</tr>
<tr>
<td>⊢ Disjunction of the guards</td>
<td>DLF</td>
</tr>
</tbody>
</table>
\[
\begin{align*}
\text{H1} \vdash G & \quad \text{HYP} \\
\frac{\text{H1}}{\text{H1}, \text{H2} \vdash G} & \quad \text{MON} \\
\frac{}{\text{P} \vdash \text{P}} & \\
\text{H, P} \vdash R & \quad \text{H, Q} \vdash R \\
\text{H, Q} \vdash R & \quad \text{OR_L} \\
\text{H, P} \vdash R & \\
\text{H, P} \vdash R \quad \text{OR_R1} \\
\text{H} \vdash \text{P} & \\
\text{H} \vdash \text{P} & \quad \text{OR_R2} \\
\text{H} \vdash \text{Q} & \\
\text{H} \vdash \text{P} \lor \text{Q} & \\
\text{H} \vdash \text{P} \lor \text{Q} & \\
\text{H} \vdash \text{P} \lor \text{Q} &
\end{align*}
\]
Equality Inference Rules

\[
\frac{H_F, E = F \vdash P_F}{H_E, E = F \vdash P_E} \quad \text{EQ\_LR}
\]

\[
\frac{H_E, E = F \vdash P_E}{H_F, E = F \vdash P_F} \quad \text{EQ\_RL}
\]
\[ \vdash 0 \in \mathbb{N} \quad \text{P1} \]
\[ n \in \mathbb{N} \vdash n + 1 \in \mathbb{N} \quad \text{P2} \]
\[ 0 < n \vdash n - 1 \in \mathbb{N} \quad \text{P2'} \]
\[ n \in \mathbb{N} \vdash 0 \leq n \quad \text{P3} \]
\[ n \leq m \vdash n - 1 \leq m \quad \text{DEC} \]
\[ n < m \vdash n + 1 \leq m \quad \text{INC} \]
Summary of Initial Model

constant: $d$

variable: $n$

axm0.1: $d \in \mathbb{N}$

axm0.2: $d > 0$

inv0.1: $n \in \mathbb{N}$

inv0.2: $n \leq d$

init
$n := 0$

ML\_out
\begin{align*}
&\text{when} \\
&\quad n < d \\
&\quad \text{then} \\
&\quad \quad n := n + 1 \\
&\quad \text{end}
\end{align*}

ML\_in
\begin{align*}
&\text{when} \\
&\quad 0 < n \\
&\quad \text{then} \\
&\quad \quad n := n - 1 \\
&\quad \text{end}
\end{align*}
- **Initial model**: Limiting the number of cars (FUN_2)

- **First refinement**: Introducing the one way bridge (FUN_3)

- **Second refinement**: Introducing the traffic lights (EQP_1,2,3)

- **Third refinement**: Introducing the sensors (EQP_4,5)
Reminder of the physical system

Island

traffic light

Bridge

Mainland

sensor
- We go down with our parachute

- Our view of the system gets more accurate

- We introduce the bridge and separate it from the island

- We shall refine the state and the events

- But we add also two new events: IL_in and IL_out

- We are focusing on FUN-3 (one way bridge)
First Refinement: Introducing a one Way Bridge
Introducing Three New Variables: $a$, $b$, and $c$

- $a$ denotes the number of cars on bridge going to island
- $b$ denotes the number of cars on island
- $c$ denotes the number of cars on bridge going to mainland
- $a$, $b$, and $c$ are the concrete variables
- They replace the abstract variable $n$
- Variables $a$, $b$, and $c$ denote natural numbers

\[
\begin{align*}
a & \in \mathbb{N} \\
b & \in \mathbb{N} \\
c & \in \mathbb{N}
\end{align*}
\]
- Relating the concrete state \((a, b, c)\) to the abstract state \((n)\)

\[
a + b + c = n
\]

- Formalizing the new invariant: one way bridge (this is FUN-3)

\[
a = 0 \lor c = 0
\]
Refining the State: Summary

constants: $d$

variables: $a, b, c$

- Invariants $\text{inv1}_1$ to $\text{inv1}_5$ are called the concrete invariants

- $\text{inv1}_4$ glues the abstract state, $n$, to the concrete state, $a, b, c$
Proposal for Refining Event \texttt{ML\_out}

\begin{center}
\begin{tikzpicture}
\node (a) at (0,0) [circle,draw] {a};
\node (b) at (0,-3) [draw] {ML\_out \begin{array}{l}
\textbf{when} \\
\quad a + b < d \\
\quad c = 0 \\
\textbf{then} \\
\quad a := a + 1 \\
\textbf{end}\end{array}};
\node (c) at (0,0) [circle,draw,fill=white] {a};
\draw[->] (a) edge (b);
\end{tikzpicture}
\end{center}
ML_in

when
0 < c
then
 c := c - 1
end
ML\_out

when

\[ a + b < d \]

\[ c = 0 \]

then

\[ a := a + 1 \]

end

ML\_in

when

\[ 0 < c \]

then

\[ c := c - 1 \]

end

Before-after predicates showing the unmodified variables:

\[ a' = a + 1 \land b' = b \land c' = c \]

\[ a' = a \land b' = b \land c' = c - 1 \]
- The concrete version is **not contradictory** with the abstract one
- When the **concrete** version is enabled then so is the abstract one
- Executions seems to be **compatible**
Intuition about refinement (2)

(abstract)ML_in

\[
\begin{align*}
\text{when} & \quad 0 < n \\
\text{then} & \quad n := n - 1
\end{align*}
\]

end

(concrete)ML_in

\[
\begin{align*}
\text{when} & \quad 0 < c \\
\text{then} & \quad c := c - 1
\end{align*}
\]

end

- Same remarks as in the previous slide

- But this has to be confirmed by well defined proof obligations
What is to be proved

- The concrete guard is stronger than the abstract one

- The actions are compatible
Constants $c$ with axioms $A(c)$

Abstract variables $\nu$ with abstract invariant $I(c, \nu)$

Concrete variables $w$ with concrete invariant $J(c, \nu, w)$

Abstract event with guards $G(c, \nu)$: $G_1(c, \nu), G_2(c, \nu), \ldots$

Abstract event with before-after predicate $\nu' = E(c, \nu)$

Concrete event with guards $H(c, w)$ and b-a predicate $w' = F(c, w)$
Axioms
Abstract Invariant
Concrete Invariant
Concrete Guard

\[ \vdash G_i(c, v) \]

<table>
<thead>
<tr>
<th>Axioms</th>
<th>\begin{align*} A(c) \ I(c, v) \ J(c, v, w) \ H(c, w) \end{align*}</th>
<th>GRD</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>[ G_i(c, v) ]</td>
<td></td>
</tr>
</tbody>
</table>
What we Have to Prove

- ML_out / GRD
- ML_in / GRD
Applying Guard Strengthening to Event ML_out

axm0_1
axm0_2
inv0_1
inv0_2
inv1_1
inv1_2
inv1_3
inv1_4
inv1_5
Concrete guards of ML_out

\[ d \in \mathbb{N} \]
\[ 0 < d \]
\[ n \in \mathbb{N} \]
\[ n \leq d \]
\[ a \in \mathbb{N} \]
\[ b \in \mathbb{N} \]
\[ c \in \mathbb{N} \]
\[ a + b + c = n \]
\[ a = 0 \lor c = 0 \]
\[ a + b < d \]
\[ c = 0 \]
\[ n < d \]

ML_out / GRD

(abstract-)ML_out
when
  \[ n < d \]
then
  \[ n := n + 1 \]
end

(concrete-)ML_out
when
  \[ a + b < d \]
  \[ c = 0 \]
then
  \[ a := a + 1 \]
end
Proof of Guard Strengthening of Event ML\_out

\[ d \in \mathbb{N} \]
\[ 0 < d \]
\[ n \in \mathbb{N} \]
\[ n \leq d \]
\[ a \in \mathbb{N} \]
\[ b \in \mathbb{N} \]
\[ c \in \mathbb{N} \]
\[ a + b + c = n \]
\[ a = 0 \lor c = 0 \]
\[ a + b < d \]
\[ c = 0 \]
\[ \vdash n < d \]

\[ a + b = n \]
\[ a + b < d \]
\[ \vdash n < d \]

\[ a + b + 0 = n \]
\[ a + b < d \]
\[ \vdash n < d \]

We use the "rule" ARITH to stand for simple arithmetic simplifications (where underlined)
Applying Guard Strengthening to Event ML_in

axm0_1
axm0_2
inv0_1
inv0_2
inv1_1
inv1_2
inv1_3
inv1_4
inv1_5
Concrete guard of ML_in
\[ d \in \mathbb{N} \]
\[ 0 < d \]
\[ n \in \mathbb{N} \]
\[ n \leq d \]
\[ a \in \mathbb{N} \]
\[ b \in \mathbb{N} \]
\[ c \in \mathbb{N} \]
\[ a + b + c = n \]
\[ a = 0 \lor c = 0 \]
\[ 0 < c \]
\[ 0 < n \]

Abstract guard of ML_in
\( \vdash \)

\( (\text{abstract-})\text{ML}_\text{in} \)
\text{when} \n\text{then} \n\text{end}

\( (\text{concrete-})\text{ML}_\text{in} \)
\text{when} \n\text{then} \n\text{end}
Proof of Guard Strengthening of Event ML\textsubscript{in}

\[
d \in \mathbb{N} \\
0 < d \\
n \in \mathbb{N} \\
n \leq d \\
a \in \mathbb{N} \\
b \in \mathbb{N} \\
c \in \mathbb{N} \\
a + b + c = n \\
a = 0 \lor c = 0 \\
0 < c \\
\vdash \\
0 < n
\]

\[
\begin{align*}
b & \in \mathbb{N} \\
a + b + c & = n \\
a & = 0 \lor c = 0 \\
0 & < c \\
\vdash \\
0 & < n
\end{align*}
\]

\[
\begin{align*}
b & \in \mathbb{N} \\
a + b + c & = n \\
a & = 0 \\
0 & < c \\
\vdash \\
0 & < n
\end{align*}
\]

\[
\begin{align*}
b & \in \mathbb{N} \\
a + b + c & = n \\
c & = 0 \\
0 & < c \\
\vdash \\
0 & < n
\end{align*}
\]

\[
\begin{align*}
b & \in \mathbb{N} \\
a + b + c & = n \\
0 & < c \\
\vdash \\
0 & < n
\end{align*}
\]

\[
\begin{align*}
b & \in \mathbb{N} \\
a + b + c & = n \\
0 & < c \\
\vdash \\
0 & < n
\end{align*}
\]

\[
\begin{align*}
b & \in \mathbb{N} \\
a + b + c & = n \\
0 & < c \\
\vdash \\
0 & < n
\end{align*}
\]
Proof of Guard Strengthening of Event ML\_in

\[
\begin{align*}
\forall d \in \mathbb{N} & \quad 0 < d \\
\forall n \in \mathbb{N} & \quad n \leq d \\
\forall a \in \mathbb{N} & \quad b \in \mathbb{N} \\
\forall c \in \mathbb{N} & \quad a + b + c = n \\
\forall a \in \mathbb{N} & \quad a = 0 \lor c = 0 \\
\forall 0 < c & \quad 0 < n \\
\end{align*}
\]

\[
\begin{align*}
\forall b \in \mathbb{N} & \quad a + b + c = n \\
\forall a \in \mathbb{N} & \quad a = 0 \lor c = 0 \\
\forall 0 < c & \quad 0 < n \\
\end{align*}
\]

\[
\begin{align*}
\forall b \in \mathbb{N} & \quad a + b + c = n \\
\forall c \in \mathbb{N} & \quad c = 0 \\
\forall 0 < c & \quad 0 < n \\
\end{align*}
\]

\[
\begin{align*}
\forall b \in \mathbb{N} & \quad a + b + c = n \\
\forall 0 < c & \quad 0 < n \\
\end{align*}
\]

\[
\begin{align*}
\forall b \in \mathbb{N} & \quad a + b + c = n \\
\forall 0 < c & \quad 0 < n \\
\end{align*}
\]
An Additional Rule: the Contradiction Rule

- In the previous proof, we have used an additional inference rule.

- It says that a false hypothesis entails any goal.

\[ \bot \vdash P \]

\[ \text{CNTR} \]
Proving Correct Refinement: Invariant Refinement

Abstract Event

\[ G(c,v) \]

I(v)

v

v' \equiv E(c,v)

J(c,v,w)

Concrete Event

\[ H(c,w) \]

w

w' \equiv F(c,w)

J(c,v',w')

I(v')
Invariant Refinement Rule

Axioms
Abstract Invariants
Concrete Invariants
Concrete Guards

\[\vdash\]

Modified Concrete Invariant

<table>
<thead>
<tr>
<th>A(c)</th>
<th>(I(c, v))</th>
<th>(J(c, v, w))</th>
<th>(H(c, w))</th>
<th>(J_j(c, E(c, v), F(c, w)))</th>
</tr>
</thead>
</table>

INV
What we Have to Prove

- ML_out / GRD done

- ML_in / GRD done

- ML_out / inv1_4 / INV

- ML_out / inv1_5 / INV

- ML_in / inv1_4 / INV

- ML_in / inv1_5 / INV
axm0.1
axm0.2
inv0.1
inv0.2
inv1.1
inv1.2
inv1.3
inv1.4
inv1.5
Concrete guards of ML\_out

\[
\begin{align*}
&d \in \mathbb{N} \\
&0 < d \\
&n \in \mathbb{N} \\
&n \leq d \\
&a \in \mathbb{N} \\
&b \in \mathbb{N} \\
&c \in \mathbb{N} \\
&a + b + c = n \\
&\text{if } a + b < d \text{ and } c = 0 \text{ then }
&n := n + 1
\end{align*}
\]

\[
\begin{align*}
&\text{ML\_out / } \text{inv1.4 / INV}
\end{align*}
\]
\begin{proof}
\begin{equation*}
d \in \mathbb{N}
\end{equation*}
\begin{equation*}
0 < d
\end{equation*}
\begin{equation*}
n \in \mathbb{N}
\end{equation*}
\begin{equation*}
n \leq d
\end{equation*}
\begin{equation*}
a \in \mathbb{N}
\end{equation*}
\begin{equation*}
b \in \mathbb{N}
\end{equation*}
\begin{equation*}
c \in \mathbb{N}
\end{equation*}
\begin{equation*}
a + b + c = n
\end{equation*}
\begin{equation*}
a = 0 \lor c = 0
\end{equation*}
\begin{equation*}
a + b < d
\end{equation*}
\begin{equation*}
c = 0
\end{equation*}
\begin{equation*}
\vdash a + 1 + b + c = n + 1
\end{equation*}
\begin{equation*}
a + b + c = n
\end{equation*}
\begin{equation*}
\vdash a + 1 + b + c = n + 1
\end{equation*}
\begin{equation*}
a + b + c = n
\end{equation*}
\begin{equation*}
\vdash a + 1 + b + c + 1 = n + 1
\end{equation*}
\begin{equation*}
\vdash n + 1 = n + 1
\end{equation*}
\end{proof}
An Additional Rule: the Equality Rule

- In the previous proof, we have used an additional inference rule:

\[ \vdash E = E \quad \text{EQL} \]

- It says that a **any term is equal to itself**

- Remark: we have not yet formally defined "term" and "predicate"

- This will be done in later lectures
Applying Invariant Refinement to Event ML_out

\[
\begin{align*}
\text{axm0}_1 & \\
\text{axm0}_2 & \\
\text{inv0}_1 & \\
\text{inv0}_2 & \\
\text{inv1}_1 & \\
\text{inv1}_2 & \\
\text{inv1}_3 & \\
\text{inv1}_4 & \\
\text{inv1}_5 & \\
\text{Concrete guards of ML_out} & \\
\vdash & \\
\text{Modified Invariant inv1}_5 & \\
\end{align*}
\]

\[
\begin{align*}
d & \in \mathbb{N} \\
0 & < d \\
n & \in \mathbb{N} \\
n & \leq d \\
a & \in \mathbb{N} \\
b & \in \mathbb{N} \\
c & \in \mathbb{N} \\
a + b + c & = n \\
a & = 0 \lor c & = 0 \\
a + b & < d \\
c & = 0 \\
\vdash & \\
a + 1 & = 0 \lor c & = 0
\end{align*}
\]

(abstract-)ML_out
\[
\begin{align*}
\text{when} & \\
n & < d \\
\text{then} & \\
n & := n + 1 \\
\text{end}
\end{align*}
\]

(concrete-)ML_out
\[
\begin{align*}
\text{when} & \\
a + b & < d \\
c & = 0 \\
\text{then} & \\
a & := a + 1 \\
\text{end}
\end{align*}
\]
Proof

$d \in \mathbb{N}$
$0 < d$
$n \in \mathbb{N}$
$n \leq d$
$a \in \mathbb{N}$
$b \in \mathbb{N}$
$c \in \mathbb{N}$
$a + b + c = n$
$a = 0 \lor c = 0$
$a + b < d$
$c = 0$
\[\vdash a + 1 = 0 \lor c = 0\]

MON

OR_R2

HYP

\[
\begin{align*}
c &= 0 \\
\vdash a + 1 &= 0 \lor c = 0
\end{align*}
\]
Applying Invariant Refinement to Event ML_in

\[
\begin{align*}
\text{axm0.1} & \quad d \in \mathbb{N} \\
\text{axm0.2} & \quad 0 < d \\
\text{inv0.1} & \quad n \in \mathbb{N} \\
\text{inv0.2} & \quad n \leq d \\
\text{inv1.1} & \quad a \in \mathbb{N} \\
\text{inv1.2} & \quad b \in \mathbb{N} \\
\text{inv1.3} & \quad c \in \mathbb{N} \\
\text{inv1.4} & \quad a + b + c = n \\
\text{inv1.5} & \quad a = 0 \lor c = 0 \\
\end{align*}
\]

Concrete guards of ML_in

\[
\begin{align*}
\Gamma \vdash & \quad a + b + c - 1 = n - 1 \\
\end{align*}
\]

Modified Invariant \textbf{inv1.4}

ML_in / \textbf{inv1.4} / INV

(abstract-)ML_in

\[
\begin{align*}
\text{when} & \quad 0 < n \\
\text{then} & \quad n := n - 1 \\
\text{end} \\
\end{align*}
\]

(concrtre-)ML_in

\[
\begin{align*}
\text{when} & \quad 0 < c \\
\text{then} & \quad c := c - 1 \\
\text{end} \\
\end{align*}
\]
Proof

\[
\begin{align*}
\begin{array}{ll}
\text{MON} & \\
\end{array}
\begin{array}{l}
d \in \mathbb{N} \\
0 < d \\
n \in \mathbb{N} \\
n \leq d \\
a \in \mathbb{N} \\
b \in \mathbb{N} \\
c \in \mathbb{N} \\
a + b + c = n \\
a = 0 \lor c = 0 \\
0 < c \\
\end{array}
\end{align*}
\]
axm0.1
axm0.2
inv0_1
inv0.2
inv1_1
inv1.2
inv1.3
inv1.4
inv1.5
Concrete guards of ML_in
\[\vdash \text{Modified Invariant } \text{inv1}_5\]

\[\text{ML_in } / \text{inv1}_5 / \text{INV}\]

\[d \in \mathbb{N}\]
\[0 < d\]
\[n \in \mathbb{N}\]
\[n \leq d\]
\[a \in \mathbb{N}\]
\[b \in \mathbb{N}\]
\[c \in \mathbb{N}\]
\[a + b + c = n\]
\[a = 0 \lor c = 0\]
\[0 < c\]
\[\vdash a = 0 \lor c - 1 = 0\]

(abstract-)ML_in
\[\text{when} \]
\[0 < n\]
\[\text{then}\]
\[n := n - 1\]
\[\text{end}\]

(concrete-)ML_in
\[\text{when} \]
\[0 < c\]
\[\text{then}\]
\[c := c - 1\]
\[\text{end}\]
\[ d \in \mathbb{N} \]
\[ 0 < d \]
\[ n \in \mathbb{N} \]
\[ n \leq d \]
\[ a \in \mathbb{N} \]
\[ b \in \mathbb{N} \]
\[ c \in \mathbb{N} \]
\[ a + b + c = n \]
\[ a = 0 \lor c = 0 \]
\[ 0 < c \]
\[ \vdash a = 0 \lor c - 1 = 0 \]
\[
d \in \mathbb{N} \\
0 < d \\
n \in \mathbb{N} \\
n \leq d \\
a \in \mathbb{N} \\
b \in \mathbb{N} \\
c \in \mathbb{N} \\
a + b + c = n \\
a = 0 \lor c = 0 \\
0 < c \\
\vdash a = 0 \lor c - 1 = 0
\]

\[
a = 0 \lor c = 0 \\
0 < c \\
\vdash a = 0 \lor c - 1 = 0
\]

\[
\ldots \\
a = 0 \\
\vdash a = 0 \lor c - 1 = 0
\]

\[
\ldots \\
0 < 0 \\
\vdash a = 0 \lor -1 = 0
\]

\[
\\perp \vdash a = 0 \lor -1 = 0
\]
- Concrete initialization

\[
\text{init} \\
\begin{array}{l}
a, b, c := 0, 0, 0
\end{array}
\]

- Corresponding after predicate

\[
a' = 0 \land b' = 0 \land c' = 0
\]
Refinement Rule for Initialization

Constants $c$ with axioms $A(c)$

Concrete invariant $J(c, v, w)$

Abstract initialization with after predicate $v' = K(c)$

Concrete initialization with after predicate $w' = L(c)$

<table>
<thead>
<tr>
<th>Axioms</th>
<th>$A(c)$</th>
<th>$J_j(c, K(c), L(c))$</th>
<th>INV</th>
</tr>
</thead>
<tbody>
<tr>
<td>⊢</td>
<td>⊢</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
What we Have to Prove

- ML_out / GRD done
- ML_in / GRD done
- ML_out / inv1_4 / INV done
- ML_out / inv1_5 / INV done
- ML_in / inv1_4 / INV done
- ML_in / inv1_5 / INV done
- inv1_4 / INV
- inv1_5 / INV
Applying the Initialization Refinement Rule

\[
\begin{align*}
\text{axm0}_1 \\
\text{axm0}_2 \\
\vdash \\
\text{Modified concrete invariant inv1}_4 \\
(a + b + c = n)
\end{align*}
\]

\[
\begin{align*}
\text{axm0}_1 \\
\text{axm0}_2 \\
\vdash \\
\text{Modified concrete invariant inv1}_5 \\
(a = 0 \lor c = 0)
\end{align*}
\]

\[
\begin{align*}
d \in \mathbb{N} \\
d > 0 \\
\vdash \\
0 + 0 + 0 = 0
\end{align*}
\]

\[
\begin{align*}
d \in \mathbb{N} \\
d > 0 \\
\vdash \\
0 = 0 \lor 0 = 0
\end{align*}
\]
New Event $\text{IL\_in}$

\begin{align*}
\text{IL\_in} \\
\text{when} \\
0 < a \\
\text{then} \\
a, b := a - 1, b + 1 \\
\text{end}
\end{align*}
New Event IL\_out

IL\_out

\textbf{when}

\begin{align*}
0 &< b \\
\alpha &= 0
\end{align*}

\textbf{then}

\begin{align*}
b, c &:= b - 1, c + 1
\end{align*}

\textbf{end}
Several Actions Done Together

Before-after predicates

\[ a' = a + 1 \quad \land \quad b' = b + 1 \quad \land \quad c' = c \]

\[ a' = a \quad \land \quad b' = b - 1 \quad \land \quad c' = c + 1 \]
The empty assignment \textit{skip}

The before-after predicate of \textit{skip} in the initial model

\[ n' = n \]

The before-after predicate of \textit{skip} in the first refinement

\[ a' = a \land b' = b \land c' = c \]
Proving Correctness of New Events

(1) A new event must refine an implicit event, made of a skip action.

(2) The new events must not diverge.
   - For this one has to exhibit a variant.
   - The variant is a natural number (could be more complicated).
   - Each new event must decrease this variant.
What we Have to Prove

- ML_out / GRD done
- ML_in / GRD done
- ML_out / inv1_4 / INV done
- ML_out / inv1_5 / INV done
- ML_in / inv1_4 / INV done
- ML_in / inv1_5 / INV done
- inv1_4 / INV done
- inv1_5 / INV done
- IL_in / inv1_4 / INV
- IL_in / inv1_5 / INV
- IL_out / inv1_4 / INV
- IL_out / inv1_5 / INV
Event \( \text{IL\_in} \) Refines \( \text{skip} \) (1)

\[
\begin{align*}
\text{axm0\_1} \\
\text{axm0\_2} \\
\text{inv0\_1} \\
\text{inv0\_2} \\
\text{inv1\_1} \\
\text{inv1\_2} \\
\text{inv1\_3} \\
\text{inv1\_4} \\
\text{inv1\_5}
\end{align*}
\]

Concrete guards of \( \text{IL\_in} \)

\[\vdash\]

Modified Invariant \( \text{inv1\_4} \)

\[
\begin{align*}
d & \in \mathbb{N} \\
0 & < d \\
n & \in \mathbb{N} \\
n & \leq d \\
a & \in \mathbb{N} \\
b & \in \mathbb{N} \\
c & \in \mathbb{N} \\
a + b + c & = n \\
a & = 0 \quad \lor \quad c & = 0 \\
0 & < a \\
\vdash \\
a - 1 + b + 1 + c & = n
\end{align*}
\]

\[
\begin{align*}
\text{IL\_in} \\
\quad \text{when} \\
\quad 0 & < a \\
\quad \text{then} \\
\quad a & := a - 1 \\
\quad b & := b + 1 \\
\quad \text{end}
\end{align*}
\]
\[
d \in \mathbb{N} \\
0 < d \\
n \in \mathbb{N} \\
n \leq d \\
a \in \mathbb{N} \\
b \in \mathbb{N} \\
c \in \mathbb{N} \\
a + b + c = n \\
a = 0 \lor c = 0 \\
0 < a \\
\vdash a - 1 + b + 1 + c = n \\
\]
Event \texttt{IL\_in} Refines \texttt{skip} (2)

\begin{align*}
\texttt{axm0\_1} \\
\texttt{axm0\_2} \\
\texttt{inv0\_1} \\
\texttt{inv0\_2} \\
\texttt{inv1\_1} \\
\texttt{inv1\_2} \\
\texttt{inv1\_3} \\
\texttt{inv1\_4} \\
\texttt{inv1\_5} \\
\text{Concrete guards of } \texttt{IL\_in} \\
\vdash \\
\text{Modified Invariant } \texttt{inv1\_5} \\
\end{align*}

\[d \in \mathbb{N}\]
\[0 < d\]
\[n \in \mathbb{N}\]
\[n \leq d\]
\[a \in \mathbb{N}\]
\[b \in \mathbb{N}\]
\[c \in \mathbb{N}\]
\[a + b + c = n\]
\[a = 0 \lor c = 0\]
\[0 < a\]
\[\vdash\]
\[a - 1 = 0 \lor c = 0\]

\begin{align*}
\texttt{IL\_in} & \quad / \quad \texttt{inv1\_5} \quad / \quad \text{INV} \\
\text{when} & \quad 0 < a \\
\text{then} & \quad a := a - 1 \\
& \quad b := b + 1 \\
\text{end} \\
\end{align*}
Proof

\begin{align*}
d \in \mathbb{N} \\
0 < d \\
n \in \mathbb{N} \\
n \leq d \\
a \in \mathbb{N} \\
b \in \mathbb{N} \\
c \in \mathbb{N} \\
a + b + c = n \\
a = 0 \lor c = 0 \\
0 < a \\
\vdash a - 1 = 0 \lor c = 0
\end{align*}

\begin{align*}
\text{MON} \\
a = 0 \lor c = 0 \\
0 < a \\
\vdash a - 1 = 0 \lor c = 0
\end{align*}

\begin{align*}
\text{OR}_L \\
\cdots
\end{align*}
Proof

\[
\begin{align*}
    d & \in \mathbb{N} \\
    0 & < d \\
    n & \in \mathbb{N} \\
    n & \leq d \\
    a & \in \mathbb{N} \\
    b & \in \mathbb{N} \\
    c & \in \mathbb{N} \\
    a + b + c & = n \\
    a & = 0 \lor c = 0 \\
    0 & < a \\
    \vdash & a - 1 = 0 \lor c = 0 \\
\end{align*}
\]

\[
\begin{align*}
    0 & < 0 \\
    \vdash & -1 = 0 \lor c = 0 \\
    c & = 0 \\
    0 & < a \\
    \vdash & a - 1 = 0 \lor c = 0 \\
\end{align*}
\]

\[
\begin{align*}
    c & = 0 \\
    \vdash & c = 0
\end{align*}
\]

\[
\begin{align*}
    c & = 0 \\
    \vdash & c = 0
\end{align*}
\]
Axioms $A(c)$, invariants $I(c, v)$, concrete invariant $J(c, v, w)$

New event with guard $H(c, w)$

Variant $V(c, w)$
Axioms $A(c)$, invariants $I(c, v)$, concrete invariant $J(c, v, w)$

New event with guard $H(c, w)$ and b-a predicate $w' = F(c, w)$

Variant $V(c, w)$

<table>
<thead>
<tr>
<th>Axioms</th>
<th>Abstract invariant</th>
<th>Concrete invariant</th>
<th>Concrete guard</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A(c)$</td>
<td>$I(c, v)$</td>
<td>$J(c, v, w)$</td>
<td>$H(c, w)$</td>
</tr>
<tr>
<td>$\vdash$</td>
<td>$\vdash$</td>
<td>$\vdash$</td>
<td>$\vdash$</td>
</tr>
<tr>
<td>Modified Var. $&lt; \text{Var.}$</td>
<td>$V(c, F(c, w)) &lt; V(c, w)$</td>
<td></td>
<td>VAR</td>
</tr>
</tbody>
</table>
variant 1: $2 \ast a + b$
What we Have to Prove

- ML_out / GRD done
- ML_in / GRD done
- ML_out / inv1_4 / INV done
- ML_out / inv1_5 / INV done
- ML_in / inv1_4 / INV done
- ML_in / inv1_5 / INV done
- inv1_4 / INV done
- inv1_5 / INV done
- IL_in / inv1_4 / INV done
- IL_in / inv1_5 / INV done
- IL_out / inv1_4 / INV done
- IL_out / inv1_5 / INV done

- NAT
- IL_in / VAR
- IL_out / VAR
Decreasing of the Variant by Event IL\_in

\begin{align*}
\text{axm0\_1} & \quad d \in \mathbb{N} \\
\text{axm0\_2} & \quad 0 < d \\
\text{inv0\_1} & \quad n \in \mathbb{N} \\
\text{inv0\_2} & \quad n \leq d \\
\text{inv1\_1} & \quad a \in \mathbb{N} \\
\text{inv1\_2} & \quad b \in \mathbb{N} \\
\text{inv1\_3} & \quad c \in \mathbb{N} \\
\text{inv1\_4} & \quad a + b + c = n \\
\text{inv1\_5} & \quad a = 0 \lor c = 0 \\
\end{align*}

Concrete guard of IL\_in

\[
\Gamma \quad \vdash \quad \text{Modified variant} < \text{Variant} \\
\]

\[
\Gamma \quad \vdash \quad 2 \ast (a - 1) + b + 1 < 2 \ast a + b
\]
Decreasing of the Variant by Event IL\_out

\[
\begin{align*}
\text{axm0\_1} \\
\text{axm0\_2} \\
\text{inv0\_1} \\
\text{inv0\_2} \\
\text{inv1\_1} \\
\text{inv1\_2} \\
\text{inv1\_3} \\
\text{inv1\_4} \\
\text{inv1\_5} \\
\text{Concrete guards of IL\_out} \\
\end{align*}
\]

\[
\begin{align*}
\vdash & \\
\text{Modified variant} < \text{Variant} \\
\end{align*}
\]

\[
\begin{align*}
d & \in \mathbb{N} \\
0 & < d \\
n & \in \mathbb{N} \\
n & \leq d \\
a & \in \mathbb{N} \\
b & \in \mathbb{N} \\
c & \in \mathbb{N} \\
a + b + c &= n \\
a &= 0 \quad \lor \quad c = 0 \\
0 & < b \\
a & = 0 \\
\vdash & \\
2 * a + b - 1 & < 2 * a + b
\end{align*}
\]

\[
\begin{align*}
\text{IL\_out / VAR} \\
& \text{when} \\
& 0 < b \\
& a = 0 \\
& \text{then} \\
& b := b - 1 \\
& c := c + 1 \\
& \text{end}
\end{align*}
\]
The $G_i(c, v)$ are the abstract guards

The $H_i(c, v)$ are the concrete guards

We have to prove the following Deadlock Freedom Rule

\[
A(c) \\
I(c, v) \\
J(c, v, w) \\
G_1(c, v) \lor \ldots \lor G_m(c, v) \\
\vdash \\
H_1(c, w) \lor \ldots \lor H_n(c, w)
\]
Applying Deadlock Freeness Rule

\[ \text{Disjunction of concrete guards} \]

\[
\begin{align*}
&d \in \mathbb{N} \\
&0 < d \\
&n \in \mathbb{N} \\
&n \leq d \\
&a \in \mathbb{N} \\
&b \in \mathbb{N} \\
&c \in \mathbb{N} \\
&a + b + c = n \\
&a = 0 \lor c = 0 \\
&\vdash (a + b < d \land c = 0) \lor c > 0 \lor a > 0 \\
&\vdash (b > 0 \land a = 0)
\end{align*}
\]

\[ ML_{\text{out}} \]
\begin{align*}
\text{when} & \quad a + b < d \\
& \quad c = 0 \\
\text{then} & \quad a := a + 1
\end{align*}
\[ ML_{\text{in}} \]
\begin{align*}
\text{when} & \quad c > 0 \\
\text{then} & \quad c := c - 1
\end{align*}
\[ IL_{\text{in}} \]
\begin{align*}
\text{when} & \quad a > 0 \\
\text{then} & \quad a := a - 1 \\
& \quad b := b + 1
\end{align*}
\[ IL_{\text{out}} \]
\begin{align*}
\text{when} & \quad b > 0 \\
& \quad a = 0 \\
\text{then} & \quad b := b - 1 \\
& \quad c := c + 1
\end{align*}
\[
\frac{H, \neg P \vdash Q}{H \vdash P \lor Q} \quad \text{NEG}
\]

\[
\frac{H, P, Q \vdash R}{H, P \land Q \vdash R} \quad \text{AND}_L
\]

\[
\frac{H \vdash P \quad H \vdash Q}{H \vdash P \land Q} \quad \text{AND}_R
\]
\[ d \in \mathbb{N} \]
\[ 0 < d \]
\[ n \in \mathbb{N} \]
\[ n \leq d \]
\[ a \in \mathbb{N} \]
\[ b \in \mathbb{N} \]
\[ c \in \mathbb{N} \]
\[ a + b + c = n \]
\[ a = 0 \lor c = 0 \]
\[ (a + b < d \land c = 0) \lor c > 0 \lor a > 0 \lor (b > 0 \land a = 0) \]

\[ 0 < d \]
\[ a \in \mathbb{N} \]
\[ b \in \mathbb{N} \]
\[ c \in \mathbb{N} \]
\[ \neg (c > 0) \]
\[ (a + b < d \land c = 0) \lor a > 0 \lor (b > 0 \land a = 0) \]
\[
\begin{align*}
&d \in \mathbb{N} \\
&0 < d \\
&n \in \mathbb{N} \\
&n \leq d \\
&a \in \mathbb{N} \\
&b \in \mathbb{N} \\
&c \in \mathbb{N} \\
&a + b + c = n \quad &\vdash (a + b < d \land c = 0) \vee \\
&a = 0 \vee &c = 0 \quad &c > 0 \vee \\
&b > 0 \vee &a > 0 \vee \\
&(b > 0 \land a = 0)
\end{align*}
\]
\[
\begin{align*}
\text{Proof (cont’d)}
\end{align*}
\]

\[
\begin{align*}
0 < d \\
\underline{a = 0} \\
\underline{b \in \mathbb{N}} \\
\vdash (a + b < d \land 0 = 0) \lor (b > 0 \land a = 0)
\end{align*}
\]

\[
\begin{align*}
0 < d \\
\underline{b \in \mathbb{N}} \\
\vdash (0 + b < d \land 0 = 0) \lor (b > 0 \land 0 = 0)
\end{align*}
\]

\[
\begin{align*}
0 < d \\
\underline{b = 0 \lor b > 0} \\
\vdash (b < d \land 0 = 0) \lor (b > 0 \land 0 = 0)
\end{align*}
\]
Proof (cont’d)

\[ 0 < d \]
\[ a = 0 \]
\[ b \in \mathbb{N} \]
\[ \vdash (a + b < d \land 0 = 0) \lor (b > 0 \land a = 0) \]

EQ_LR

\[ 0 < d \]
\[ b \in \mathbb{N} \]
\[ \vdash (0 + b < d \land 0 = 0) \lor (b > 0 \land 0 = 0) \]

ARITH

\[ 0 < d \]
\[ b = 0 \lor b > 0 \]
\[ \vdash (b < d \land 0 = 0) \lor (b > 0 \land 0 = 0) \]

OR_L

\[ \{ \]
\[ 0 < d \]
\[ b = 0 \]
\[ \vdash b < d \land 0 = 0 \]

OR_R1

\[ 0 < d \]
\[ b > 0 \]
\[ \vdash (b < d \land 0 = 0) \lor (b > 0 \land 0 = 0) \]

OR_R2

\[ \} \]

\[ \{ \]
\[ 0 < d \]
\[ \vdash 0 = 0 \]

HYP

\[ 0 < d \]
\[ \vdash 0 < d \]

EQL

\[ b > 0 \]
\[ \vdash b > 0 \]

HYP

\[ b > 0 \]
\[ \vdash 0 = 0 \]

EQL
### What we Proved

- ML\_out / GRD done
- ML\_in / GRD done
- ML\_out / inv1\_4 / INV done
- ML\_out / inv1\_5 / INV done
- ML\_in / inv1\_4 / INV done
- ML\_in / inv1\_5 / INV done
- inv1\_4 / INV done
- inv1\_5 / INV done
- IL\_in / inv1\_4 / INV done
- IL\_in / inv1\_5 / INV done
- IL\_out / inv1\_4 / INV done
- IL\_out / inv1\_5 / INV done
- NAT done
- IL\_in / VAR done
- IL\_out / VAR done
- DLF done
What we Proved

- For old events:
  - Strengthening of guards: rule GRD
  - Concrete invariant preservation: rule INV

- For new events:
  - Refining the implicit skip event: rule INV
  - Absence of divergence: rules NAT and VAR

- For all events:
  - Relative deadlock freeness: rule DLF
### Refinement Rules (1)

<table>
<thead>
<tr>
<th>Axioms</th>
<th>GRD</th>
<th>Axioms</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstract invariants</td>
<td></td>
<td>Abstract invariants</td>
</tr>
<tr>
<td>Concrete invariants</td>
<td></td>
<td>Concrete invariants</td>
</tr>
<tr>
<td>Concrete guards</td>
<td></td>
<td>Concrete guard</td>
</tr>
<tr>
<td>Abstract guard</td>
<td></td>
<td>Modified concrete invariant</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Axioms</th>
<th>INV</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Modified concrete invariant</td>
<td></td>
</tr>
</tbody>
</table>
Refinement Rules (2)

\[
\begin{array}{c|c}
\text{Axioms} & \text{NAT} \\
\text{Abstract invariants} & \\
\text{Concrete invariants} & \\
\text{Concrete guards of a new event} & \\
\vdash & \\
\text{Variant} \in \mathbb{N} & \\
\end{array}
\]

\[
\begin{array}{c|c}
\text{Axioms} & \text{VAR} \\
\text{Abstract invariants} & \\
\text{Concrete invariants} & \\
\text{Concrete guards of a new event} & \\
\vdash & \\
\text{Modified variant} < \text{Variant} & \\
\end{array}
\]

\[
\begin{array}{c|c}
\text{Axioms} & \text{DLF} \\
\text{Abstract invariants} & \\
\text{Concrete invariants} & \\
\text{Disjunction of abstract events guards} & \\
\vdash & \\
\text{Disjunction of concrete events guards} & \\
\end{array}
\]
variables: $a, b, c$

inv1_1: $a \in \mathbb{N}$

inv1_2: $b \in \mathbb{N}$

inv1_3: $c \in \mathbb{N}$

inv1_4: $a + b + c = n$

inv1_5: $a = 0 \lor c = 0$

variant1: $2 \ast a + b$
Events of the First Refinement

init

\[
\begin{align*}
    a & := 0 \\
    b & := 0 \\
    c & := 0
\end{align*}
\]

ML_in

\[
\begin{align*}
    \text{when} & \quad 0 < c \\
    \text{then} & \quad c := c - 1
\end{align*}
\]

ML_out

\[
\begin{align*}
    \text{when} & \quad a + b < d \\
    \quad & \quad c = 0 \\
    \text{then} & \quad a := a + 1
\end{align*}
\]

IL_in

\[
\begin{align*}
    \text{when} & \quad 0 < a \\
    \text{then} & \quad a := a - 1 \\
    & \quad b := b + 1
\end{align*}
\]

IL_out

\[
\begin{align*}
    \text{when} & \quad 0 < b \\
    & \quad a = 0 \\
    \text{then} & \quad b := b - 1 \\
    & \quad c := c + 1
\end{align*}
\]
- **Initial model**: Limiting the number of cars (FUN_2)

- **First refinement**: Introducing the one way bridge (FUN_3)

- **Second refinement**: Introducing the traffic lights (EQP_1,2,3)

- **Third refinement**: Introducing the sensors (EQP_4,5)
Second Refinement: Introducing Traffic Lights
Extending the Constants

set: \( COLOR \)

constants: \( red, green \)

\[
\text{axm2.1: } COLOR = \{green, \text{red}\}
\]

\[
\text{axm2.2: } green \neq \text{red}
\]
Extending the Variables

\[ il_{\text{tl}} \in \text{COLOR} \quad \text{and} \quad ml_{\text{tl}} \in \text{COLOR} \]

Remark: Events IL_{\text{in}} and ML_{\text{in}} are not modified in this refinement
- A green mainland traffic light implies safe access to the bridge
- A green mainland traffic light implies safe access to the bridge

\[ ml_{tl} = \text{green} \implies c = 0 \land a + b < d \]
Refining Event ML_out

(abstract)ML_out

when
  c = 0
  a + b < d
then
  a := a + 1
end
Refining Event ML_out

(abstract)ML_out

when
   c = 0
   a + b < d
then
   a := a + 1
end

(concrete)ML_out

when
   ml_tl = green
then
   a := a + 1
end
- A green island traffic light implies safe access to the bridge
- A green island traffic light implies safe access to the bridge

\[ il_{tl} = \text{green} \Rightarrow a = 0 \land 0 < b \]
Refining Event IL_out

(abstract) IL_out

    when
      \( a = 0 \)
      \( 0 < b \)
    then
      \( b, c := b - 1, c + 1 \)
    end
(abstract) IL_out

when
   a = 0
   0 < b
then
   b, c := b − 1, c + 1
end

(concrete) IL_out

when
   il_tl = green
then
   b, c := b − 1, c + 1
end
New Events $ML_{tl\_green}$ and $IL_{tl\_green}$

- Turning lights to green when proper conditions hold

ML_{tl\_green}

\textbf{when} \quad ml\_tl = \text{red} \\
\quad c = 0 \\
\quad a + b < d \\
\textbf{then} \\
\quad ml\_tl := \text{green} \\
\textbf{end}

IL_{tl\_green}

\textbf{when} \quad il\_tl = \text{red} \\
\quad a = 0 \\
\quad 0 < b \\
\textbf{then} \\
\quad il\_tl := \text{green} \\
\textbf{end}
Summary of State Refinement (so far)

variables: $a, b, c, ml_{tl}, il_{tl}$

inv2_1: $ml_{tl} \in \text{COLOR}$

inv2_2: $il_{tl} \in \text{COLOR}$

inv2_3: $ml_{tl} = \text{green} \Rightarrow a + b < d \land c = 0$

inv2_4: $il_{tl} = \text{green} \Rightarrow 0 < b \land a = 0$
Summary of Old Events (so far)

ML_out
when
ml_tl = green
then
    a := a + 1
end

IL_out
when
il_tl = green
then
    b := b - 1
    c := c + 1
end

Events ML_in and IL_in are unchanged

ML_in
when
0 < c
then
    c := c - 1
end

IL_in
when
0 < a
then
    a := a - 1
    b := b + 1
end
variables:  \(a, b, c, ml\_tl, il\_tl\)

- Variables \(a, b,\) and \(c\) were present in the previous refinement.

- Variables \(ml\_tl\) and \(il\_tl\) are superposed to \(a, b,\) and \(c\).

- We have thus to extend rule INV.
Abstract Event
\[
\text{when } G(c, u, v) \\
\text{then } \\
\quad u := E(c, u, v) \\
\quad v := M(c, u, v) \\
\text{end}
\]

Concrete Event
\[
\text{when } H(c, v, w) \\
\text{then } \\
\quad v := N(c, v, w) \\
\quad w := F(c, v, w) \\
\text{end}
\]

Axioms
- Abstract invariants
- Concrete invariants
- Concrete guards

⇒
- Same actions on common variables

\[
A(c) \\
I(c, u, v) \\
J(c, u, v, w) \\
H(c, v, w) \\
\Rightarrow \quad M(c, u, v) = N(c, v, w)
\]

SIM
- We have to apply 3 Proof Obligations:
  - GRD,
  - SIM,
  - INV

- On 4 events: ML_out, IL_out, ML_in, IL_in

- And 2 main invariants:

\[
\text{inv2.3: } ml_{tl} = \text{green} \Rightarrow a + b < d \land c = 0 \\
\text{inv2.4: } il_{tl} = \text{green} \Rightarrow 0 < b \land a = 0
\]
Proving the Refinement of the Four Old Events

- SIM is completely trivial since the actions are the same
- GRD is also trivial

\[
\text{inv2.3: } ml_{tl} = \text{green} \implies a + b < d \land c = 0
\]

\[
\text{inv2.4: } il_{tl} = \text{green} \implies 0 < b \land a = 0
\]
Proving the Refinement of the Four Old Events

- INV applied to ML_in and IL_in holds trivially

\[
\text{inv2.3: } \quad ml_tl = \text{green} \implies a + b < d \land c = 0
\]

\[
\text{inv2.4: } \quad il_tl = \text{green} \implies 0 < b \land a = 0
\]
Proving the Refinement of the Four Old Events

\[
\text{ML\_out} \\
\begin{array}{l}
\text{when} \\
\quad c = 0 \\
\quad a + b < d \\
\text{then} \\
\quad a := a + 1 \\
\text{end}
\end{array}
\]

\[
\text{IL\_out} \\
\begin{array}{l}
\text{when} \\
\quad a = 0 \\
\quad 0 < b \\
\text{then} \\
\quad b := b - 1 \\
\quad c := c + 1 \\
\text{end}
\end{array}
\]

\[
\text{ML\_in} \\
\begin{array}{l}
\text{when} \\
\quad 0 < c \\
\text{then} \\
\quad c := c - 1 \\
\text{end}
\end{array}
\]

\[
\text{IL\_in} \\
\begin{array}{l}
\text{when} \\
\quad 0 < a \\
\text{then} \\
\quad a := a - 1 \\
\quad b := b + 1 \\
\text{end}
\end{array}
\]

\[
\text{ML\_out} \\
\begin{array}{l}
\text{when} \\
\quad ml\_tl = \text{green} \\
\text{then} \\
\quad a := a + 1 \\
\text{end}
\end{array}
\]

\[
\text{IL\_out} \\
\begin{array}{l}
\text{when} \\
\quad il\_tl = \text{green} \\
\text{then} \\
\quad b := b - 1 \\
\quad c := c + 1 \\
\text{end}
\end{array}
\]

\[
\text{ML\_in} \\
\begin{array}{l}
\text{when} \\
\quad 0 < c \\
\text{then} \\
\quad c := c - 1 \\
\text{end}
\end{array}
\]

\[
\text{IL\_in} \\
\begin{array}{l}
\text{when} \\
\quad 0 < a \\
\text{then} \\
\quad a := a - 1 \\
\quad b := b + 1 \\
\text{end}
\end{array}
\]

- INV applied to ML\_out and IL\_out raise some difficulties
- ML_out / inv2_4 / INV

- IL_out / inv2_3 / INV

- ML_out / inv2_3 / INV

- IL_out / inv2_4 / INV
More Logical Rules of Inferences

- Rules about implication

\[
\begin{align*}
H, P, Q & \vdash R & \text{IMP}_L \\
H, P, P \Rightarrow Q & \vdash R \\
H & \vdash P \Rightarrow Q & \text{IMP}_R
\end{align*}
\]

- Rules about negation

\[
\begin{align*}
H & \vdash P & \text{NOT}_L \\
H, \neg P & \vdash Q \\
H, P & \vdash \neg Q & \text{NOT}_R
\end{align*}
\]
axm0_1
axm0_2
axm2_1
axm2_2
inv0_1
inv0_2
inv1_1
inv1_2
inv1_3
inv1_4
inv1_5
inv2_1
inv2_2
inv2_3
inv2_4

Guard of event ML_out

\[ d \in \mathbb{N} \]
\[ 0 < d \]
\[ \text{COLOR} = \{\text{green}, \text{red}\} \]
\[ \text{green} \neq \text{red} \]
\[ n \in \mathbb{N} \]
\[ n \leq d \]
\[ a \in \mathbb{N} \]
\[ b \in \mathbb{N} \]
\[ c \in \mathbb{N} \]
\[ a + b + c = n \]
\[ a = 0 \lor c = 0 \]
\[ ml\_tl \in \text{COLOR} \]
\[ il\_tl \in \text{COLOR} \]
\[ ml\_tl = \text{green} \Rightarrow a + b < d \land c = 0 \]
\[ il\_tl = \text{green} \Rightarrow 0 < b \land a = 0 \]
\[ ml\_tl = \text{green} \]
\[ il\_tl = \text{green} \Rightarrow 0 < b \land a + 1 = 0 \]

Modified invariant inv2_4

ML_out / inv2_4 / INV

when
  ml_tl = green
then
  a := a + 1
end
\[
d \in \mathbb{N} \\
0 < d \\
\text{COLOR} = \{\text{green, red}\} \\
\text{green} \neq \text{red} \\
\color{red}n \in \mathbb{N} \\
\color{red}n \leq d \\
\color{red}a \in \mathbb{N} \\
\color{red}b \in \mathbb{N} \\
\color{red}c \in \mathbb{N} \\
\color{red}a + b + c = n \\
\color{red}a = 0 \lor \color{red}c = 0 \\
\color{red}ml_{tl} \in \text{COLOR} \\
\color{red}il_{tl} \in \text{COLOR} \\
\color{red}ml_{tl} = \text{green} \Rightarrow a + b < d \land c = 0 \\
\color{red}il_{tl} = \text{green} \Rightarrow 0 < b \land a = 0 \\
\color{red}ml_{tl} = \text{green} \\
\vdash \color{red}il_{tl} = \text{green} \Rightarrow 0 < b \land a + 1 = 0
\]

\[
\text{green} \neq \text{red} \\
\color{red}il_{tl} = \text{green} \Rightarrow 0 < b \land a = 0 \\
\color{red}ml_{tl} = \text{green} \\
\vdash \color{red}il_{tl} = \text{green} \Rightarrow 0 < b \land a + 1 = 0
\]
\[
d \in \mathbb{N}
0 < d
COLOR = \{\text{green, red}\}
green \neq \text{red}
n \in \mathbb{N}
n \leq d
a \in \mathbb{N}
b \in \mathbb{N}
c \in \mathbb{N}
a + b + c = n
a = 0 \lor c = 0
ml_{tl} \in \text{COLOR}
il_{tl} \in \text{COLOR}
ml_{tl} = \text{green} \Rightarrow a + b < d \land c = 0
il_{tl} = \text{green} \Rightarrow 0 < b \land a = 0
ml_{tl} = \text{green}
\vdash il_{tl} = \text{green} \Rightarrow 0 < b \land a + 1 = 0
\]

\[
green \neq \text{red}
il_{tl} = \text{green} \Rightarrow 0 < b \land a = 0
ml_{tl} = \text{green}
\vdash il_{tl} = \text{green} \Rightarrow 0 < b \land a + 1 = 0
\]

\[
\text{green} \neq \text{red}
il_{tl} = \text{green} \Rightarrow 0 < b \land a = 0
ml_{tl} = \text{green}
il_{tl} = \text{green}
\vdash 0 < b \land a + 1 = 0
\]

\[
\text{green} \neq \text{red}
il_{tl} = \text{green} \Rightarrow 0 < b \land a = 0
ml_{tl} = \text{green}
il_{tl} = \text{green}
\vdash 0 < b \land a + 1 = 0
\]
Tentative Proof (cont’d)

... 

\[
\begin{align*}
green \neq \text{red} \\
0 < b \\
a = 0 \\
ml\_tl = \text{green} \\
il\_tl = \text{green} \\
\vdash 0 < b \\
a + 1 = 0
\end{align*}
\]

\[
\begin{align*}
green \neq \text{red} \\
0 < b \\
a = 0 \\
ml\_tl = \text{green} \\
il\_tl = \text{green} \\
\vdash \quad 0 < b \\
a + 1 = 0
\end{align*}
\]

\[
\begin{align*}
green \neq \text{red} \\
0 < b \\
a = 0 \\
ml\_tl = \text{green} \\
il\_tl = \text{green} \\
\vdash \quad 0 + 1 = 0
\end{align*}
\]

\[
\begin{align*}
green \neq \text{red} \\
ml\_tl = \text{green} \\
il\_tl = \text{green} \\
\vdash \quad 1 = 0
\end{align*}
\]
Proving Preservation of inv2.3 by Event IL_out

\begin{align*}
axm0.1 & \quad d \in \mathbb{N} \\
axm0.2 & \quad 0 < d \\
axm2.1 & \quad COLOR = \{\text{green, red}\} \\
axm2.2 & \quad \text{green} \neq \text{red} \\
inv0.1 & \quad n \in \mathbb{N} \\
inv0.2 & \quad n \leq d \\
inv1.1 & \quad a \in \mathbb{N} \\
inv1.2 & \quad b \in \mathbb{N} \\
inv1.3 & \quad c \in \mathbb{N} \\
inv1.4 & \quad a + b + c = n \\
inv1.5 & \quad a = 0 \quad \lor \quad c = 0 \\
inv2.1 & \quad ml_tl \in \text{COLOR} \\
inv2.2 & \quad il_tl \in \text{COLOR} \\
inv2.3 & \quad ml_tl = \text{green} \implies a + b < d \quad \land \quad c = 0 \\
inv2.4 & \quad il_tl = \text{green} \implies 0 < b \quad \land \quad a = 0 \\
\text{Guard of IL_out} & \quad il_tl = \text{green} \\
\vdash & \quad ml_tl = \text{green} \implies a + b - 1 < d \quad \land \quad c + 1 = 0
\end{align*}

\begin{align*}
\text{IL_out} & \quad / \quad \text{inv2.3} \quad / \quad \text{INV} \\
& \quad \text{when} \\
& \quad \quad il_tl = \text{green} \\
& \quad \text{then} \\
& \quad \quad b := b - 1 \\
& \quad \quad c := c + 1 \\
& \quad \text{end}
\end{align*}
\[ d \in \mathbb{N} \]
\[ 0 < d \]
\[ \text{COLOR} = \{\text{green, red}\} \]
\[ \text{green} \neq \text{red} \]
\[ n \in \mathbb{N} \]
\[ n \leq d \]
\[ a \in \mathbb{N} \]
\[ b \in \mathbb{N} \]
\[ c \in \mathbb{N} \]
\[ a + b + c = n \]
\[ a = 0 \lor c = 0 \]
\[ ml_{tl} \in \text{COLOR} \]
\[ il_{tl} \in \text{COLOR} \]
\[ ml_{tl} = \text{green} \Rightarrow a + b < d \land c = 0 \]
\[ il_{tl} = \text{green} \Rightarrow 0 < b \land a = 0 \]
\[ \vdash ml_{tl} = \text{green} \Rightarrow a + b - 1 < d \land c + 1 = 0 \]
\begin{align*}
d & \in \mathbb{N} \\
0 & < d \\
COLOR & = \{\text{green, red}\} \\
\text{green} & \neq \text{red} \\
n & \in \mathbb{N} \\
n & \leq d \\
a & \in \mathbb{N} \\
b & \in \mathbb{N} \\
c & \in \mathbb{N} \\
 a + b + c & = n \\
a & = 0 \lor c = 0 \\
ml_{tl} & \in \text{COLOR} \\
il_{tl} & \in \text{COLOR} \\
ml_{tl} = \text{green} & \Rightarrow a + b < d \land c = 0 \\
il_{tl} = \text{green} & \Rightarrow 0 < b \land a = 0 \\
il_{tl} = \text{green} & \\
\vdash ml_{tl} = \text{green} & \Rightarrow a + b - 1 < d \land c + 1 = 0 \\
\end{align*}
...
- In both cases, we were stopped by attempting to prove the following

\[
\begin{align*}
green \neq \text{red} \\
il_{tl} &= \text{green} \\
ml_{tl} &= \text{green}
\end{align*}
\]
\[\vdash 1 = 0\]

Both traffic lights are assumed to be green!

- This indicates that an "obvious" invariant was missing

- In fact, at least one of the two traffic lights must be red

\[
\text{inv2.5: } ml_{tl} = \text{red} \lor il_{tl} = \text{red}
\]
Completing the Proof

\[
\text{green} \neq \text{red} \\
\text{ml}_tl = \text{red} \quad \lor \quad \text{il}_tl = \text{red} \\
\text{il}_tl = \text{green} \\
\text{ml}_tl = \text{green} \\
\vdash 1 = 0
\]

\[
\text{green} \neq \text{red} \\
\text{il}_tl = \text{red} \\
\text{il}_tl = \text{green} \\
\text{ml}_tl = \text{green} \\
\vdash 1 = 0
\]

\[
\text{green} \neq \text{red} \\
\text{green} = \text{red} \\
\text{ml}_tl = \text{green} \\
\vdash 1 = 0
\]

\[
\text{green} \neq \text{red} \\
\text{green} = \text{red} \\
\text{il}_tl = \text{red} \\
\vdash 1 = 0
\]

\[
\text{green} \neq \text{red} \\
\text{green} = \text{red} \\
\text{il}_tl = \text{green} \\
\text{ml}_tl = \text{green} \\
\vdash 1 = 0
\]

\[
\text{green} = \text{red} \\
\text{il}_tl = \text{green} \\
\vdash \text{green} = \text{red}
\]
Going back to the Requirements Document

\[
\text{inv2.5: } \ ml\_tl = \text{red} \ \lor \ il\_tl = \text{red}
\]

This could have been deduced from these requirements

<table>
<thead>
<tr>
<th>FUN-3</th>
<th>The bridge is one way or the other, not both at the same time</th>
</tr>
</thead>
<tbody>
<tr>
<td>EQP-3</td>
<td>Cars are not supposed to pass on a red traffic light, only on a green one</td>
</tr>
</tbody>
</table>
What we Have to Prove

- ML_out / inv2_4 / INV done

- IL_out / inv2_3 / INV done

- ML_out / inv2_3 / INV

- IL_out / inv2_4 / INV

- ML_tl_green / inv2_5 / INV

- IL_tl_green / inv2_5 / INV
Proving Preservation of inv2.3 by Event ML_out

\begin{align*}
\text{axm0}_1 & \quad d \in \mathbb{N} \\
\text{axm0}_2 & \quad 0 < d \\
\text{axm2}_1 & \quad \text{COLOR} = \{\text{green, red}\} \\
\text{axm2}_2 & \quad \text{green} \neq \text{red} \\
\text{inv0}_1 & \quad n \in \mathbb{N} \\
\text{inv0}_2 & \quad n \leq d \\
\text{inv1}_1 & \quad a \in \mathbb{N} \\
\text{inv1}_2 & \quad b \in \mathbb{N} \\
\text{inv1}_3 & \quad c \in \mathbb{N} \\
\text{inv1}_4 & \quad a + b + c = n \\
\text{inv1}_5 & \quad a = 0 \lor c = 0 \\
\text{inv2}_1 & \quad \text{ml}_tl \in \text{COLOR} \\
\text{inv2}_2 & \quad \text{il}_tl \in \text{COLOR} \\
\text{inv2}_3 & \quad \text{ml}_tl = \text{green} \Rightarrow a + b < d \land c = 0 \\
\text{inv2}_4 & \quad \text{il}_tl = \text{green} \Rightarrow 0 < b \land a = 0 \\
\text{Guard of ML_out} & \quad \text{ml}_tl = \text{green} \Rightarrow 0 < b \land a = 0 \\
\text{ML_out / inv2.3 / INV} & \quad \text{ml}_tl = \text{green} \Rightarrow a + 1 + b < d \land c = 0
\end{align*}

```
ML_out
  when
    ml_tl = green
  then
    a := a + 1
end
```
Tentative Proof

\[ d \in \mathbb{N} \]
\[ 0 < d \]
\[ \text{COLOR} = \{\text{green}, \text{red}\} \]
\[ \text{green} \neq \text{red} \]
\[ n \in \mathbb{N} \]
\[ n \leq d \]
\[ a \in \mathbb{N} \]
\[ b \in \mathbb{N} \]
\[ c \in \mathbb{N} \]
\[ a + b + c = n \]
\[ a = 0 \lor c = 0 \]
\[ ml_{tl} \in \text{COLOR} \]
\[ il_{tl} \in \text{COLOR} \]
\[ ml_{tl} = \text{green} \Rightarrow a + b < d \land c = 0 \]
\[ il_{tl} = \text{green} \Rightarrow 0 < b \land a = 0 \]
\[ ml_{tl} = \text{green} \]
\[ \vdash ml_{tl} = \text{green} \Rightarrow a + 1 + b < d \land c = 0 \]
\begin{align*}
d \in \mathbb{N} \\
0 < d \\
COLOR = \{\text{green, red}\} \\
\text{green} \neq \text{red} \\
n \in \mathbb{N} \\
n \leq d \\
a \in \mathbb{N} \\
b \in \mathbb{N} \\
c \in \mathbb{N} \\
a + b + c = n \\
a = 0 \lor c = 0 \\
ml_{tl} \in \text{COLOR} \\
il_{tl} \in \text{COLOR} \\
ml_{tl} = \text{green} \Rightarrow a + b < d \land c = 0 \\
il_{tl} = \text{green} \Rightarrow 0 < b \land a = 0 \\
ml_{tl} = \text{green} \Rightarrow a + 1 + b < d \land c = 0 \\
\implies ml_{tl} = \text{green} \Rightarrow a + b < d \land c = 0 \land a + 1 + b < d \land c = 0
\end{align*}
- This requires splitting the ML\_out in two separate events ML\_out\_1 and ML\_out\_2

**ML\_out\_1**

```plaintext
when
  ml\_tl = green
then
  a + 1 + b < d
  a := a + 1
end
```

**ML\_out\_2**

```plaintext
when
  ml\_tl = green
then
  a + 1 + b = d
  a := a + 1
  ml\_tl := red
end
```
- When \( a + 1 + b = d \) then only one more car can enter the island.

- Consequently, the traffic light \( ml\_tl \) must be turned red (while the car enters the bridge).
Proving Preservation of \( \text{inv2}_3 \) by Event \( \text{ML}_{\text{out}}_1 \)

\[
\begin{align*}
\text{axm0}_1 & : \quad d \in \mathbb{N} \\
\text{axm0}_2 & : \quad 0 < d \\
\text{axm2}_1 & : \quad \text{COLOR} = \{\text{green}, \text{red}\} \\
\text{axm2}_2 & : \quad \text{green} \neq \text{red} \\
\text{inv0}_1 & : \quad n \in \mathbb{N} \\
\text{inv0}_2 & : \quad n \leq d \\
\text{inv1}_1 & : \quad a \in \mathbb{N} \\
\text{inv1}_2 & : \quad b \in \mathbb{N} \\
\text{inv1}_3 & : \quad c \in \mathbb{N} \\
\text{inv1}_4 & : \quad a + b + c = n \\
\text{inv1}_5 & : \quad a = 0 \lor c = 0 \\
\text{inv2}_1 & : \quad \text{ml}_tl \in \text{COLOR} \\
\text{inv2}_2 & : \quad \text{il}_tl \in \text{COLOR} \\
\text{inv2}_3 & : \quad \text{ml}_tl = \text{green} \Rightarrow a + b < d \land c = 0 \\
\text{inv2}_4 & : \quad \text{il}_tl = \text{green} \Rightarrow 0 < b \land a = 0 \\
\text{Guard of } \text{ML}_{\text{out}}_1 & : \quad \text{ml}_tl = \text{green} \Rightarrow a + 1 + b < d \\
\hline
\text{ML}_{\text{out}}_1 & : \quad \text{when } \text{ml}_tl = \text{green} \\
 & \quad a + 1 + b < d \\
 & \quad \text{then } a := a + 1 \\
& \quad \text{end}
\end{align*}
\]
Proof

\[
\begin{align*}
d & \in \mathbb{N} \\
0 & < d \\
COLOR & = \{\text{green, red}\} \\
\text{green} & \neq \text{red} \\
n & \in \mathbb{N} \\
n & \leq d \\
a & \in \mathbb{N} \\
b & \in \mathbb{N} \\
c & \in \mathbb{N} \\
a + b + c & = n \\
a & = 0 \lor c = 0 \\
ml_{tl} & \in \text{COLOR} \\
il_{tl} & \in \text{COLOR} \\
m_{tl} = \text{green} & \Rightarrow a + b < d \land c = 0 \\
ml_{tl} = \text{green} & \Rightarrow 0 < b \land a = 0 \\
m_{tl} = \text{green} & \Rightarrow a + 1 + b < d \\
\vdash & \\
m_{tl} = \text{green} & \Rightarrow a + 1 + b < d \land c = 0
\end{align*}
\]
Proof

\[
\begin{align*}
d \in \mathbb{N} \\
0 < d \\
COLOR = \{\text{green, red}\} \\
green \neq \text{red} \\
n \in \mathbb{N} \\
n \leq d \\
a \in \mathbb{N} \\
b \in \mathbb{N} \\
c \in \mathbb{N} \\
a + b + c = n \\
a = 0 \lor c = 0 \\
ml_{tl} \in \text{COLOR} \\
il_{tl} \in \text{COLOR} \\
ml_{tl} = \text{green} \Rightarrow a + b < d \land c = 0 \\
il_{tl} = \text{green} \Rightarrow 0 < b \land a = 0 \\
ml_{tl} = \text{green} \Rightarrow a + 1 + b < d \\
\vdash ml_{tl} = \text{green} \Rightarrow a + 1 + b < d \land c = 0 \\
\end{align*}
\]

\[
\begin{align*}
ml_{tl} = \text{green} \Rightarrow a + b < d \land c = 0 \\
\vdash a + 1 + b < d \\
\vdash ml_{tl} = \text{green} \\
\vdash a + 1 + b < d \\
\end{align*}
\]
Proof (cont’d)

\[ a + b < d \\
\quad c = 0 \\
\quad ml\_tl = \text{green} \\
\quad a + 1 + b < d \]
\[ \vdash a + 1 + b < d \quad \land \quad c = 0 \]
Proving Preservation of inv2\_3 by Event ML\_out\_2

\[
\begin{aligned}
\text{axm0}\_1 & \quad d \in \mathbb{N} \\
\text{axm0}\_2 & \quad 0 < d \\
\text{axm2}\_1 & \quad \text{COLOR} = \{\text{green}, \text{red}\} \\
\text{axm2}\_2 & \quad \text{green} \neq \text{red} \\
\text{inv0}\_1 & \quad n \in \mathbb{N} \\
\text{inv0}\_2 & \quad n \leq d \\
\text{inv1}\_1 & \quad a \in \mathbb{N} \\
\text{inv1}\_2 & \quad b \in \mathbb{N} \\
\text{inv1}\_3 & \quad c \in \mathbb{N} \\
\text{inv1}\_4 & \quad a + b + c = n \\
\text{inv1}\_5 & \quad a = 0 \quad \vee \quad c = 0 \\
\text{inv2}\_1 & \quad \text{ml}\_tl \in \text{COLOR} \\
\text{inv2}\_2 & \quad \text{il}\_tl \in \text{COLOR} \\
\text{inv2}\_3 & \quad \text{ml}\_tl = \text{green} \Rightarrow a + b < d \quad \land \quad c = 0 \\
\text{inv2}\_4 & \quad \text{il}\_tl = \text{green} \Rightarrow 0 < b \quad \land \quad a = 0 \\
\text{Guard of ML}\_\text{out}\_2 & \quad \text{ml}\_tl = \text{green} \\
& \quad a + 1 + b = d \\
\end{aligned}
\]

\[
\begin{aligned}
\text{ML}\_\text{out}\_2 & / \text{inv2}\_3 / \text{INV} \\
\vdash & \quad \text{modified inv2}\_3 \\
& \quad \text{red} = \text{green} \Rightarrow a + 1 + b < d \quad \land \quad c = 0 \\
\end{aligned}
\]

\[
\begin{aligned}
\text{ML}\_\text{out}\_2 & \\
& \quad \text{when} \\
& \quad \text{ml}\_tl = \text{green} \\
& \quad \text{a} + 1 + b = d \\
& \quad \text{then} \\
& \quad \text{a} := \text{a} + 1 \\
& \quad \text{ml}\_tl := \text{red} \\
& \quad \text{end}
\end{aligned}
\]
\[
\begin{align*}
d &\in \mathbb{N} \\
0 &< d \\
\text{COLOR} &= \{\text{green}, \text{red}\} \\
\text{green} &\neq \text{red} \\
n &\in \mathbb{N} \\
n &\leq d \\
a &\in \mathbb{N} \\
b &\in \mathbb{N} \\
c &\in \mathbb{N} \\
a + b + c &= n \\
a &= 0 \lor c = 0 \\
ml_{tl} &\in \text{COLOR} \\
il_{tl} &\in \text{COLOR} \\
ml_{tl} = \text{green} &\Rightarrow a + b < d \land c = 0 \\
il_{tl} = \text{green} &\Rightarrow 0 < b \land a = 0 \\
ml_{tl} = \text{green} \\
a + 1 + b &= d \\
\therefore \\
\text{red} = \text{green} &\Rightarrow a + 1 + b < d \land c = 0
\end{align*}
\]
Proof

\[
\begin{align*}
d & \in \mathbb{N} \\
0 & < d \\
COLOR & = \{\text{green, red}\} \\
\text{green} & \neq \text{red} \\
n & \in \mathbb{N} \\
n & \leq d \\
a & \in \mathbb{N} \\
b & \in \mathbb{N} \\
c & \in \mathbb{N} \\
a + b + c & = n \\
a & = 0 \lor c = 0 \\
ml_{tl} & \in \text{COLOR} \\
il_{tl} & \in \text{COLOR} \\
ml_{tl} = \text{green} & \Rightarrow a + b < d \land c = 0 \\
il_{tl} = \text{green} & \Rightarrow 0 < b \land a = 0 \\
ml_{tl} = \text{green} & \Rightarrow a + 1 + b = d \\
\vdash & \\
\text{red} = \text{green} & \Rightarrow a + 1 + b < d \land c = 0
\end{align*}
\]

\[
\begin{align*}
\text{green} & \neq \text{red} \\
\vdash & \\
\text{red} & = \text{green} \Rightarrow a + 1 + b < d \land c = 0
\end{align*}
\]

\[
\begin{align*}
\vdash & \\
\text{green} & \neq \text{green} \\
a + 1 + b & < d \land c = 0
\end{align*}
\]

\[
\begin{align*}
\vdash & \\
\text{green} & = \text{green}
\end{align*}
\]
What we Have to Prove

- ML_out / \textbf{inv2.4} / INV done

- IL_out / \textbf{inv2.3} / INV done

- ML_out / \textbf{inv2.3} / INV done

- IL_out / \textbf{inv2.4} / INV

- ML_tl_green / \textbf{inv2.5} / INV

- IL_tl_green / \textbf{inv2.5} / INV
Proving Preservation of inv2_4 by Event \texttt{IL\_out}

\begin{align*}
\text{axm0\_1} & \\
\text{axm0\_2} & \\
\text{axm2\_1} & \\
\text{axm2\_2} & \\
\text{inv0\_1} & \\
\text{inv0\_2} & \\
\text{inv1\_1} & \\
\text{inv1\_2} & \\
\text{inv1\_3} & \\
\text{inv1\_4} & \\
\text{inv1\_5} & \\
\text{inv2\_1} & \\
\text{inv2\_2} & \\
\text{inv2\_3} & \\
\text{inv2\_4} & \\
\text{Guard of event IL\_out} &
\end{align*}

\begin{align*}
\text{Modified invariant \texttt{inv2\_4}}
\end{align*}

\begin{align*}
d & \in \mathbb{N} \\
0 & < d \\
\text{COLOR} & = \{\text{green, red}\} \\
\text{green} & \neq \text{red} \\
n & \in \mathbb{N} \\
n & \leq d \\
a & \in \mathbb{N} \\
b & \in \mathbb{N} \\
c & \in \mathbb{N} \\
a + b + c & = n \\
a & = 0 \lor c & = 0 \\
\text{ml\_tl} & \in \text{COLOR} \\
\text{il\_tl} & \in \text{COLOR} \\
\text{ml\_tl} & = \text{green} \Rightarrow a + b < d \land c = 0 \\
\text{il\_tl} & = \text{green} \Rightarrow 0 < b \land a = 0 \\
\text{il\_tl} & = \text{green} \\
\vdash & \\
\text{IL\_out} & / \text{inv2\_4} / \text{INV} \ \text{when} \ \text{il\_tl} = \text{green} \ \text{then} \ b : = b - 1 \ \text{end}
\end{align*}
\( d \in \mathbb{N} \)
\( 0 < d \)
\( \text{COLOR} = \{\text{green, red}\} \)
\( \text{green} \neq \text{red} \)
\( n \in \mathbb{N} \)
\( n \leq d \)
\( a \in \mathbb{N} \)
\( b \in \mathbb{N} \)
\( c \in \mathbb{N} \)
\( a + b + c = n \)
\( a = 0 \lor c = 0 \)
\( ml_{tl} \in \text{COLOR} \)
\( il_{tl} \in \text{COLOR} \)
\( ml_{tl} = \text{green} \Rightarrow a + b < d \land c = 0 \)
\( il_{tl} = \text{green} \Rightarrow 0 < b \land a = 0 \)
\( il_{tl} = \text{green} \)
\( \vdash il_{tl} = \text{green} \Rightarrow 0 < b - 1 \land a = 0 \)

\( 0 < b \land a = 0 \)
\( \vdash 0 < b - 1 \land a = 0 \)
Tentative Proof (cont’d)

\[ 0 < b \]
\[ a = 0 \]
\[ \vdash 0 < b - 1 \]

\[ 0 < b \]
\[ a = 0 \]
\[ \vdash a = 0 \]
\[ \text{HYP} \]

AND_R \{ 
\[ 0 < b \]
\[ a = 0 \]
\[ \vdash 0 < b - 1 \]
\[ \text{MON} \]
\[ a = 0 \]
\[ \vdash a = 0 \]
\[ \text{MON} \]
\[ 0 < b \]
\[ a = 0 \]
\[ \vdash 0 < b - 1 \]
\[ ? \]

- This requires splitting the concrete IL_out in two separate events IL_out_1 and IL_out_2

IL_out_1
\[ \text{when} \]
\[ \text{il}_\text{tl} = \text{green} \]
\[ b \neq 1 \]
\[ \text{then} \]
\[ b, c := b - 1, c + 1 \]
\[ \text{end} \]

IL_out_2
\[ \text{when} \]
\[ \text{il}_\text{tl} = \text{green} \]
\[ b = 1 \]
\[ \text{then} \]
\[ b, c := b - 1, c + 1 \]
\[ \text{il}_\text{tl} := \text{red} \]
\[ \text{end} \]
IL_out_1
  when
  il_tl = green
  b \neq 1
  then
  b, c := b - 1, c + 1
  end

IL_out_2
  when
  il_tl = green
  b = 1
  then
  b, c := b - 1, c + 1
  il_tl := red
  end

- When b=1, then only one car remains in the island

- Consequently, the traffic light il_tl can be turned red
  (after this car has left)
Proving Preservation of inv2.4 by Event IL_out_1

\[
\begin{align*}
    \text{d} & \in \mathbb{N} \\
    0 & < d \\
    \text{COLOR} & = \{\text{green}, \text{red}\} \\
    \text{green} & \neq \text{red} \\
    n & \in \mathbb{N} \\
    n & \leq d \\
    a & \in \mathbb{N} \\
    b & \in \mathbb{N} \\
    c & \in \mathbb{N} \\
    a + b + c & = n \\
    a & = 0 \lor c = 0 \\
    ml_tl & \in \text{COLOR} \\
    il_tl & \in \text{COLOR} \\
    ml_tl & = \text{green} \Rightarrow a + b < d \land c = 0 \\
    il_tl & = \text{green} \Rightarrow 0 < b \land a = 0 \\
    il_tl & = \text{green} \Rightarrow 0 < b - 1 \land a = 0 \\
    \end{align*}
\]

Guard of event IL_out_1

\(\vdash\) Modified invariant inv2.4

\begin{verbatim}
IL_out_1
when
    il_tl = green
    b ≠ 1
then
    b, c := b - 1, c + 1
end
\end{verbatim}
Proof

\[ d \in \mathbb{N} \]
\[ 0 < d \]
\[ \text{COLOR} = \{\text{green, red}\} \]
\[ \text{green} \neq \text{red} \]
\[ n \in \mathbb{N} \]
\[ n \leq d \]
\[ a \in \mathbb{N} \]
\[ b \in \mathbb{N} \]
\[ c \in \mathbb{N} \]
\[ a + b + c = n \]
\[ a = 0 \lor c = 0 \]
\[ ml_{tl} \in \text{COLOR} \]
\[ il_{tl} \in \text{COLOR} \]
\[ ml_{tl} = \text{green} \Rightarrow a + b < d \land c = 0 \]
\[ il_{tl} = \text{green} \Rightarrow 0 < b \land a = 0 \]
\[ il_{tl} = \text{green} \Rightarrow 0 < b - 1 \land a = 0 \]
\[ ml_{tl} = \text{green} \Rightarrow 0 < b \land a = 0 \]
\[ b \neq 1 \]
\[ il_{tl} = \text{green} \Rightarrow 0 < b - 1 \land a = 0 \]

...
Proof (cont’d)

\[ 0 < b \]
\[ a = 0 \]
\[ b \neq 1 \]
\[ \vdash 0 < b - 1 \wedge a = 0 \]

\[ 0 < b \]
\[ a = 0 \]
\[ b \neq 1 \]
\[ \vdash 0 < b - 1 \]

\[ 0 < b \]
\[ a = 0 \]
\[ b \neq 1 \]
\[ \vdash a = 0 \]

\[ 0 < b - 1 \]
\[ b \neq 1 \]
\[ \vdash 0 < b - 1 \]

\[ 0 < b - 1 \]
\[ \vdash 0 < b - 1 \]
Proving Preservation of inv2.4 by Event IL_out_2

\begin{align*}
\text{axm0.1} & \quad d \in \mathbb{N} \\
\text{axm0.2} & \quad 0 < d \\
\text{axm2.1} & \quad \text{COLOR} = \{\text{green}, \text{red}\} \\
\text{axm2.2} & \quad \text{green} \neq \text{red} \\
\text{inv0.1} & \quad n \in \mathbb{N} \\
\text{inv0.2} & \quad n \leq d \\
\text{inv1.1} & \quad a \in \mathbb{N} \\
\text{inv1.2} & \quad b \in \mathbb{N} \\
\text{inv1.3} & \quad c \in \mathbb{N} \\
\text{inv1.4} & \quad a + b + c = n \\
\text{inv1.5} & \quad a = 0 \lor c = 0 \\
\text{inv2.1} & \quad ml_tl \in \text{COLOR} \\
\text{inv2.2} & \quad il_tl \in \text{COLOR} \\
\text{inv2.3} & \quad ml_tl = \text{green} \implies a + b < d \land c = 0 \\
\text{il_tl} = \text{green} \implies 0 < b \land a = 0 \\
\text{il_tl} = \text{green} \implies b = 1 \\
\text{red} = \text{green} \implies 0 < b - 1 \land a = 0
\end{align*}

Guard of event IL_out_2

\[ \text{IL_out_2} \]
\[ \text{when} \]
\[ il_tl = \text{green} \]
\[ b = 1 \]
\[ \text{then} \]
\[ b, c, il_tl := b - 1, c + 1, red \]
\[ \text{end} \]
Proof

\[ d \in \mathbb{N} \]
\[ 0 < d \]
\[ COLOR = \{\text{green, red}\} \]
\[ \text{green} \neq \text{red} \]
\[ n \in \mathbb{N} \]
\[ n \leq d \]
\[ a \in \mathbb{N} \]
\[ b \in \mathbb{N} \]
\[ c \in \mathbb{N} \]
\[ a + b + c = n \]
\[ a = 0 \lor c = 0 \]
\[ ml.tl \in COLOR \]
\[ il.tl \in COLOR \]
\[ ml.tl = \text{green} \Rightarrow a + b < d \land c = 0 \]
\[ il.tl = \text{green} \Rightarrow 0 < b \land a = 0 \]
\[ b = 1 \]
\[ \vdash \text{red} = \text{green} \Rightarrow 0 < b - 1 \land a = 0 \]

\[ \vdash \text{green} \neq \text{red} \]
\[ \vdash \text{red} = \text{green} \Rightarrow 0 < b - 1 \land a = 0 \]

\[ \vdash 0 < b - 1 \land a = 0 \]

\[ \vdash \text{green} \neq \text{green} \]
\[ \vdash 0 < b - 1 \land a = 0 \]

\[ \vdash \text{green} = \text{green} \]
What we Have to Prove

- ML_out / \texttt{inv2_4} / INV done

- IL_out / \texttt{inv2_3} / INV done

- ML_out / \texttt{inv2_3} / INV done

- IL_out / \texttt{inv2_4} / INV done

- ML_tl_green / \texttt{inv2_5} / INV

- IL_tl_green / \texttt{inv2_5} / INV
But the new invariant inv2.5 is not preserved by the new events

\[
\text{inv2.5: } \ml_{tl} = \text{red} \lor \il_{tl} = \text{red}
\]

Unless we correct them as follows:

\[
\begin{align*}
\text{ML}_{tl}\text{.green} & \quad \text{when} \\
& \quad \ml_{tl} = \text{red} \\
& \quad a + b < d \\
& \quad c = 0 \\
& \quad \text{then} \\
& \quad \ml_{tl} := \text{green} \\
& \quad \il_{tl} := \text{red} \\
& \quad \text{end} \\
\end{align*}
\]

\[
\begin{align*}
\text{IL}_{tl}\text{.green} & \quad \text{when} \\
& \quad \il_{tl} = \text{red} \\
& \quad 0 < b \\
& \quad a = 0 \\
& \quad \text{then} \\
& \quad \il_{tl} := \text{green} \\
& \quad \ml_{tl} := \text{red} \\
& \quad \text{end}
\end{align*}
\]
Summary of the Proof Situation

- Correct event refinement:  **OK**

- Absence of divergence of new events:  **FAILURE**

- Absence of deadlock:  **?**
When $a$ and $c$ are both equal to 0 and $b$ is positive, then both events are always alternatively enabled.

The lights can change colors very rapidly.
ML\_tl\_green and IL\_tl\_green can run for ever
ML_tl_green and IL_tl_green can run for ever
ML_tl_green and IL_tl_green can run for ever
ML_tl_green and IL_tl_green can run for ever
ML\_tl\_green and IL\_tl\_green can run for ever
ML_tl_green and IL_tl_green can run for ever
- Allowing each light to turn green only when at least one car has passed in the other direction

- For this, we introduce two additional variables:

\[
\begin{align*}
\text{inv2}_6: & \quad ml\_pass \in \{0, 1\} \\
\text{inv2}_7: & \quad il\_pass \in \{0, 1\}
\end{align*}
\]
Modifying Events ML_out_1 and ML_out_2

ML_out_1

when
  ml_tl = green
  a + 1 + b < d
then
  a := a + 1
  ml_pass := 1
end

ML_out_2

when
  ml_tl = green
  a + 1 + b = d
then
  a := a + 1
  ml_tl := red
  ml_pass := 1
end
IL\_out\_1
    when
        il\_tl = green
        b \neq 1
    then
        b ::= b - 1
        c ::= c + 1
        il\_pass ::= 1
    end

IL\_out\_2
    when
        il\_tl = green
        b = 1
    then
        b ::= b - 1
        c ::= c + 1
        il\_tl ::= red
        il\_pass ::= 1
    end
Modifying Events ML\_tl\_gree and IL\_tl\_green

\[
\text{ML\_tl\_gree}
\begin{align*}
\text{when} & \quad ml\_tl = \text{red} \\
& \quad a + b < d \\
& \quad c = 0 \\
& \quad il\_pass = 1 \\
\text{then} & \quad ml\_tl := \text{green} \\
& \quad il\_tl := \text{red} \\
& \quad ml\_pass := 0 \\
\text{end}
\end{align*}
\]

\[
\text{IL\_tl\_gree}
\begin{align*}
\text{when} & \quad il\_tl = \text{red} \\
& \quad 0 < b \\
& \quad a = 0 \\
& \quad ml\_pass = 1 \\
\text{then} & \quad il\_tl := \text{green} \\
& \quad ml\_tl := \text{red} \\
& \quad il\_pass := 0 \\
\text{end}
\end{align*}
\]

\[
\text{end}
\]

We exhibit the following variant

\begin{boxedenv}
\textbf{variant 2:} \quad ml\_pass + il\_pass
\end{boxedenv}
To be Proved

\[
\begin{align*}
ml_{tl} &= \text{red} \\
a + b &< d \\
c &= 0 \\
il_{pass} &= 1 \\
\Rightarrow &&
\begin{align*}
il_{pass} + 0 &< \\
ml_{pass} + il_{pass} &< \\
\end{align*}
\end{align*}
\]

\[
\begin{align*}
il_{tl} &= \text{red} \\
b &> 0 \\
a &= 0 \\
ml_{pass} &= 1 \\
\Rightarrow &&
\begin{align*}
ml_{pass} + 0 &< \\
ml_{pass} + il_{pass} &< \\
\end{align*}
\end{align*}
\]

This cannot be proved. This suggests the following invariants:

inv2.8: \( ml_{tl} = \text{red} \Rightarrow ml_{pass} = 1 \)

inv2.9: \( il_{tl} = \text{red} \Rightarrow il_{pass} = 1 \)
\begin{align*}
0 & < d \\
ml_{tl} & \in \{\text{red, green}\} \\
il_{tl} & \in \{\text{red, green}\} \\
ml_{pass} & \in \{0, 1\} \\
il_{pass} & \in \{0, 1\} \\
a & \in \mathbb{N} \\
b & \in \mathbb{N} \\
c & \in \mathbb{N} \\
ml_{tl} = \text{red} & \Rightarrow ml_{pass} = 1 \\
il_{tl} = \text{red} & \Rightarrow il_{pass} = 1 \\
\Rightarrow & \\
( ml_{tl} = \text{red} & \land a + b < d \land c = 0 \land il_{pass} = 1 ) \lor \\
( il_{tl} = \text{red} & \land a = 0 \land b > 0 \land ml_{pass} = 1 ) \lor \\
ml_{tl} = \text{green} & \lor il_{tl} = \text{green} \lor a > 0 \lor c > 0
\end{align*}
The previous statement reduces to the following, which is true:

\[
\begin{align*}
0 < d & \quad a \in \mathbb{N} \\
b \in \mathbb{N} & \quad c \in \mathbb{N} \\
\Rightarrow & \\
(a + b < d \land c = 0) \lor (a = 0 \land b > 0) \lor a > 0 \lor c > 0
\end{align*}
\]
- Thanks to the proofs:
  - We discovered 4 errors
  - We introduced several additional invariants
  - We corrected 4 events
  - We introduced 2 more variables
### Conclusion: we Introduced the Superposition Rule

<table>
<thead>
<tr>
<th>Axioms</th>
<th>SIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstract invariants</td>
<td></td>
</tr>
<tr>
<td>Concrete invariants</td>
<td></td>
</tr>
<tr>
<td>Concrete guards</td>
<td></td>
</tr>
<tr>
<td>$\vdash$</td>
<td></td>
</tr>
<tr>
<td>Same actions on common variables</td>
<td></td>
</tr>
</tbody>
</table>
variables: \( a, b, c, ml_{tl}, il_{tl}, ml_{pass}, il_{pass} \)

inv2.1: \( ml_{tl} \in \{\text{red, green}\} \)

inv2.2: \( il_{tl} \in \{\text{red, green}\} \)

inv2.3: \( ml_{tl} = 1 \implies a + b < d \land c = 0 \)

inv2.4: \( il_{tl} = 1 \implies 0 < b \land a = 0 \)
inv2.5:  \( ml_{tl} = \text{red} \lor il_{tl} = \text{red} \)

inv2.6:  \( ml_{pass} \in \{0, 1\} \)

inv2.7:  \( il_{pass} \in \{0, 1\} \)

inv2.8:  \( ml_{tl} = \text{red} \Rightarrow ml_{pass} = 1 \)

inv2.9:  \( il_{tl} = \text{red} \Rightarrow il_{pass} = 1 \)

variant2:  \( ml_{pass} + il_{pass} \)
Summary of Second Refinement: the Event (1)

ML\_out\_1
when
  \( ml\_tl = \text{green} \)
  \( a + 1 + b < d \)
then
  \( a := a + 1 \)
  \( ml\_pass := 1 \)
end

ML\_out\_2
when
  \( ml\_tl = \text{green} \)
  \( a + 1 + b = d \)
then
  \( a := a + 1 \)
  \( ml\_pass := 1 \)
  \( ml\_tl := \text{red} \)
end
IL_out_1

when
  \( il_{tl} = \text{green} \)
  \( b \neq 1 \)
then
  \( b := b - 1 \)
  \( c := c + 1 \)
  \( il_{pass} := 1 \)
end

IL_out_2

when
  \( il_{tl} = \text{green} \)
  \( b = 1 \)
then
  \( b := b - 1 \)
  \( c := c + 1 \)
  \( il_{pass} := 1 \)
  \( il_{tl} := \text{red} \)
end
ML_tl_green

when
  ml_tl = red
  a + b < d
  c = 0
  il_pass = 1
then
  ml_tl := green
  il_tl := red
  ml_pass := 0
end

IL_tl_green

when
  il_tl = red
  0 < b
  a = 0
  ml_pass := 1
then
  il_tl := green
  ml_tl := red
  il_pass := 0
end
- These events are identical to their abstract versions

\[
\text{ML}_{\text{in}}\\
\quad \text{when} \quad 0 < c \quad \text{then} \quad c := c - 1 \quad \text{end}
\]

\[
\text{IL}_{\text{in}}\\
\quad \text{when} \quad 0 < a \quad \text{then} \quad a := a - 1 \\
\quad \quad b := b + 1 \quad \text{end}
\]
- **Initial model**: Limiting the number of cars (FUN_2)

- **First refinement**: Introducing the one way bridge (FUN_3)

- **Second refinement**: Introducing the traffic lights (EQP_1,2,3)

- **Third refinement**: Introducing the sensors (EQP_4,5)
Reminder of the physical system
- We want to clearly identify in our model:
  - The controller
  - The environment
  - The communication channels between the two
Controller variables: $a,$

$b,$

$c,$

$ml\_pass,$

$il\_pass$
These new variables denote physical objects

Environment variables: A, B, C, ML_OUT_SR, ML_IN_SR, IL_OUT_SR, IL_IN_SR
Output channels: $ml_{tl}$,

$il_{tl}$
Input channels: $ml\_out\_10$, $ml\_in\_10$, $il\_in\_10$, $il\_out\_10$

A message is sent when a sensor moves from "on" to "off":

```
off

on

off

sending a message to the controller
```
carrier sets: \ldots, Sensor

constants: \ldots, on, off

axm3_1: \[ SENSOR = \{on, off\} \]

axm3_2: \[ on \neq off \]
\begin{align*}
\text{inv3}_1 & : \quad \text{M} \text{L} \text{.} \text{O} \text{U} \text{T} \text{.} \text{S} \text{R} \in \text{S} \text{E} \text{N} \text{S} \text{O} \text{R} \\
\text{inv3}_2 & : \quad \text{M} \text{L} \text{.} \text{I} \text{N} \text{.} \text{S} \text{R} \in \text{S} \text{E} \text{N} \text{S} \text{O} \text{R} \\
\text{inv3}_3 & : \quad \text{I} \text{L} \text{.} \text{O} \text{U} \text{T} \text{.} \text{S} \text{R} \in \text{S} \text{E} \text{N} \text{S} \text{O} \text{R} \\
\text{inv3}_4 & : \quad \text{I} \text{L} \text{.} \text{I} \text{N} \text{.} \text{S} \text{R} \in \text{S} \text{E} \text{N} \text{S} \text{O} \text{R}
\end{align*}
\begin{align*}
\text{inv3\_5} & : ~ A \in \mathbb{N} \\
\text{inv3\_6} & : ~ B \in \mathbb{N} \\
\text{inv3\_7} & : ~ C \in \mathbb{N} \\
\text{inv3\_8} & : ~ ml\_out\_10 \in \text{BOOL} \\
\text{inv3\_9} & : ~ ml\_in\_10 \in \text{BOOL} \\
\text{inv3\_10} & : ~ il\_out\_10 \in \text{BOOL} \\
\text{inv3\_11} & : ~ il\_in\_10 \in \text{BOOL}
\end{align*}
Invariants (1)

When sensors are on, there are cars on them

\[ \text{inv3.12} : \quad IL\_IN\_SR = on \quad \Rightarrow \quad A > 0 \]
\[ \text{inv3.13} : \quad IL\_OUT\_SR = on \quad \Rightarrow \quad B > 0 \]
\[ \text{inv3.14} : \quad ML\_IN\_SR = on \quad \Rightarrow \quad C > 0 \]

The sensors are used to detect the presence of cars entering or leaving the bridge
Drivers obey the traffic lights

\begin{align*}
\text{inv3}_{15} : & \quad ml\_out\_10 = \text{TRUE} \quad \Rightarrow \quad ml\_tl = \text{green} \\
\text{inv3}_{16} : & \quad il\_out\_10 = \text{TRUE} \quad \Rightarrow \quad il\_tl = \text{green}
\end{align*}

Cars are not supposed to pass on a red traffic light, only on a green one

EQP-3
When a sensor is "on", the previous information is treated

\[
\begin{align*}
\text{inv3}_17 : & \quad IL\_IN\_SR = \text{on} \implies il\_in\_10 = \text{FALSE} \\
\text{inv3}_18 : & \quad IL\_OUT\_SR = \text{on} \implies il\_out\_10 = \text{FALSE} \\
\text{inv3}_19 : & \quad ML\_IN\_SR = \text{on} \implies ml\_in\_10 = \text{FALSE} \\
\text{inv3}_20 : & \quad ML\_OUT\_SR = \text{on} \implies ml\_out\_10 = \text{FALSE}
\end{align*}
\]

The controller must be fast enough so as to be able to treat all the information coming from the environment
### Linking the physical and logical cars (1)

<table>
<thead>
<tr>
<th>Invariant</th>
<th>Condition</th>
<th>Equation</th>
</tr>
</thead>
<tbody>
<tr>
<td>inv3_21</td>
<td>$il_{in\ 10} = \text{TRUE} \land ml_{out\ 10} = \text{TRUE}$</td>
<td>$A = a$</td>
</tr>
<tr>
<td>inv3_22</td>
<td>$il_{in\ 10} = \text{FALSE} \land ml_{out\ 10} = \text{TRUE}$</td>
<td>$A = a + 1$</td>
</tr>
<tr>
<td>inv3_23</td>
<td>$il_{in\ 10} = \text{TRUE} \land ml_{out\ 10} = \text{FALSE}$</td>
<td>$A = a - 1$</td>
</tr>
<tr>
<td>inv3_24</td>
<td>$il_{in\ 10} = \text{FALSE} \land ml_{out\ 10} = \text{FALSE}$</td>
<td>$A = a$</td>
</tr>
</tbody>
</table>
Linking the physical and logical cars (2)

\[
\begin{align*}
\text{inv3.25} : \quad & il\_in\_10 = \text{TRUE} \land il\_out\_10 = \text{TRUE} \quad \Rightarrow \quad B = b \\
\text{inv3.26} : \quad & il\_in\_10 = \text{TRUE} \land il\_out\_10 = \text{FALSE} \quad \Rightarrow \quad B = b + 1 \\
\text{inv3.27} : \quad & il\_in\_10 = \text{FALSE} \land il\_out\_10 = \text{TRUE} \quad \Rightarrow \quad B = b - 1 \\
\text{inv3.28} : \quad & il\_in\_10 = \text{FALSE} \land il\_out\_10 = \text{FALSE} \quad \Rightarrow \quad B = b
\end{align*}
\]

\[
\begin{align*}
\text{inv3.29} : \quad & il\_out\_10 = \text{TRUE} \land ml\_out\_10 = \text{TRUE} \quad \Rightarrow \quad C = c \\
\text{inv3.30} : \quad & il\_out\_10 = \text{TRUE} \land ml\_out\_10 = \text{FALSE} \quad \Rightarrow \quad C = c + 1 \\
\text{inv3.31} : \quad & il\_out\_10 = \text{FALSE} \land ml\_out\_10 = \text{TRUE} \quad \Rightarrow \quad C = c - 1 \\
\text{inv3.32} : \quad & il\_out\_10 = \text{FALSE} \land ml\_out\_10 = \text{FALSE} \quad \Rightarrow \quad C = c
\end{align*}
\]
The basic properties hold for the physical cars

\[
\text{inv3.33} : \ A = 0 \ \lor \ C = 0 \\
\text{inv3.34} : \ A + B + C \leq d
\]

The number of cars on the bridge and the island is limited

The bridge is one way or the other, not both at the same time

FUN-2

FUN-3
When $ml\_out\_10 = \text{TRUE}$ and $a + b + 1 \neq d$
\[
a := a + 1 \\
ml\_pass := 1 \\
ml\_out\_10 := \text{FALSE}
\]
\text{end}

When $ml\_out\_10 = \text{TRUE}$ and $a + b + 1 = d$
\[
a := a + 1 \\
ml\_tl := \text{red} \\
ml\_pass := 1 \\
ml\_out\_10 := \text{FALSE}
\]
\text{end}

When $ml\_tl = \text{green}$ and $a + b + 1 \neq d$
\[
a := a + 1 \\
ml\_pass := 1
\]
\text{end}

When $ml\_tl = \text{green}$ and $a + b + 1 = d$
\[
a := a + 1 \\
ml\_pass := 1 \\
ml\_tl := \text{red}
\]
\text{end}
IL_out_1
  when
    il_out_10 = TRUE
    b ≠ 1
  then
    b := b - 1
    c := c + 1
    il_pass := 1
    il_out_10 := FALSE
  end

IL_out_2
  when
    il_out_10 = TRUE
    b = 1
  then
    b := b - 1
    c := c + 1
    il_tl := red
    il_pass := 1
    il_out_10 := FALSE
  end

(abstract-)IL_out_1
  when
    il_tl = green
    b ≠ 1
  then
    b := b - 1
    c := c + 1
    il_pass := 1
  end

(abstract-)IL_out_2
  when
    il_tl = green
    b = 1
  then
    b := b - 1
    c := c + 1
    il_pass := 1
    il_tl := red
  end
Refining Abstract Events (3)

\[
\text{ML\_in}
\begin{align*}
\text{when} \quad & \quad ml\_in\_10 = \text{TRUE} \\
\text{then} \quad & \quad 0 < c \\
\text{then} \quad & \quad c := c - 1 \\
\text{end} \\
\text{end}
\]

\[
\text{IL\_in}
\begin{align*}
\text{when} \quad & \quad il\_in\_10 = \text{TRUE} \\
\text{then} \quad & \quad 0 < a \\
\text{then} \quad & \quad a := a - 1 \\
\text{then} \quad & \quad b := b + 1 \\
\text{end} \\
\text{end}
\]

\[
\text{(abstract-)ML\_in}
\begin{align*}
\text{when} \quad & \quad 0 < c \\
\text{then} \quad & \quad c := c - 1 \\
\text{end}
\]

\[
\text{(abstract-)IL\_in}
\begin{align*}
\text{when} \quad & \quad 0 < a \\
\text{then} \quad & \quad a := a - 1 \\
\text{then} \quad & \quad b := b + 1 \\
\text{end}
\]
Refining Abstract Events (4)

ML\_tl\_green
---
when
\( ml\_tl = \text{red} \)
\( a + b < d \)
\( c = 0 \)
\( il\_pass = 1 \)
\( il\_out\_10 = \text{FALSE} \)
then
\( ml\_tl := \text{green} \)
\( il\_tl := \text{red} \)
\( ml\_pass := \text{FALSE} \)
end

IL\_tl\_green
---
when
\( il\_tl = \text{red} \)
\( a = 0 \)
\( ml\_pass = 1 \)
\( ml\_out\_10 = \text{FALSE} \)
then
\( il\_tl := \text{green} \)
\( ml\_tl := \text{red} \)
\( il\_pass := \text{FALSE} \)
end

(abstract-)ML\_tl\_green
---
when
\( ml\_tl = \text{red} \)
\( a + b < d \)
\( c = 0 \)
\( il\_pass = 1 \)
then
\( ml\_tl := \text{green} \)
\( il\_tl := \text{red} \)
\( ml\_pass := 0 \)
end

(abstract-)IL\_tl\_green
---
when
\( il\_tl = \text{red} \)
\( 0 < b \)
\( a = 0 \)
\( ml\_pass = 1 \)
then
\( il\_tl := \text{green} \)
\( ml\_tl := \text{red} \)
\( il\_pass := 0 \)
end
Adding New PHYSICAL Events (1)

ML_out_arr
when
    \( \text{ML\_OUT\_SR} = \text{off} \)
    \( ml\_out\_10 = \text{FALSE} \)
then
    \( \text{ML\_OUT\_SR} := \text{on} \)
end

ML_in_arr
when
    \( \text{ML\_IN\_SR} = \text{off} \)
    \( ml\_in\_10 = \text{FALSE} \)
    \( C > 0 \)
then
    \( \text{ML\_IN\_SR} := \text{on} \)
end

IL_in_arr
when
    \( \text{IL\_IN\_SR} = \text{off} \)
    \( il\_in\_10 = \text{FALSE} \)
    \( A > 0 \)
then
    \( \text{IL\_IN\_SR} := \text{on} \)
end

IL_out_arr
when
    \( \text{IL\_OUT\_SR} = \text{off} \)
    \( il\_out\_10 = \text{FALSE} \)
    \( B > 0 \)
then
    \( \text{IL\_OUT\_SR} := \text{on} \)
end
Adding New PHYSICAL Events (2)

ML_out_dep
when
  ML_OUT_SR = on
  ml_tl = green
then
  ML_OUT_SR := off
  ml_out_10 := TRUE
end

ML_in_dep
when
  ML_IN_SR = on
then
  ML_IN_SR := off
  ml_in_10 := TRUE
  C = C - 1
end

IL_in_dep
when
  IL_IN_SR = on
then
  IL_IN_SR := off
  il_in_10 := TRUE
  A = A - 1
  B = B + 1
end

IL_out_dep
when
  IL_OUT_SR = on
  il_tl = green
then
  IL_OUT_SR := off
  il_out_10 := TRUE
  B = B - 1
  C = C + 1
end
Final Structure of the Controller

Constant: \( d \)
Variables: \( a, b, c, \) il\_pass, ml\_pass
ml\_in\_10
ml\_out\_10
il\_in\_10
il\_out\_10
IL\_OUT\_SR
IL\_IN\_SR
ML\_OUT\_SR
ML\_IN\_SR
ML\_IN\_SR
ML\_OUT\_SR
il\_tl
ml\_tl
8 logical Events
8 physical Events
- **What** is to be **systematically** proved?
  - **Invariant** preservation
  - **Correct refinements** of transitions
  - **No divergence** of new transitions
  - **No deadlock** introduced in refinements

- **When** are these proofs done?
Questions on Proving (cont’d)

- Who states what is to be proved?
  - An automatic tool: the Proof Obligation Generator

- Who is going to perform these proofs?
  - An automatic tool: the Prover
  - Sometimes helped by the Engineer (interactive proving)
- Three basic tools:
  - Proof Obligation Generator
  - Prover
  - Model translators into Hardware or Software languages

- These tools are embedded into a Development Data Base

- Such tools already exist in the Rodin Platform
Summary of Proofs on Example

- This development required 237 proofs
  - Initial model: 7
  - 1st refinement: 26
  - 2nd refinement: 66
  - 3rd refinement: 138

- All proved automatically by the Rodin Platform
<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P \land Q$</td>
<td>conjunction</td>
</tr>
<tr>
<td>$P \lor Q$</td>
<td>disjunction</td>
</tr>
<tr>
<td>$P \Rightarrow Q$</td>
<td>implication</td>
</tr>
<tr>
<td>$\neg P$</td>
<td>negation</td>
</tr>
<tr>
<td>$x \in S$</td>
<td>set membership operator</td>
</tr>
<tr>
<td><strong>N</strong></td>
<td>set of Natural Numbers: {0, 1, 2, 3, \ldots}</td>
</tr>
<tr>
<td>-------</td>
<td>-----------------------------------------------</td>
</tr>
<tr>
<td><strong>Z</strong></td>
<td>set of Integers: {0, 1, −1, 2, −2, \ldots}</td>
</tr>
<tr>
<td>{a, b, \ldots}</td>
<td>set defined in extension</td>
</tr>
<tr>
<td><strong>a + b</strong></td>
<td>addition of (a) and (b)</td>
</tr>
<tr>
<td><strong>a − b</strong></td>
<td>subtraction of (a) and (b)</td>
</tr>
</tbody>
</table>
### Summary of Mathematical Notations (3)

<table>
<thead>
<tr>
<th>Notation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a \ast b$</td>
<td>product of $a$ and $b$</td>
</tr>
<tr>
<td>$a = b$</td>
<td>equality relation</td>
</tr>
<tr>
<td>$a \leq b$</td>
<td>smaller than or equal relation</td>
</tr>
<tr>
<td>$a &lt; b$</td>
<td>smaller than relation</td>
</tr>
</tbody>
</table>
- For the init event in the initial model

| Axioms of the constants | \(\Rightarrow\) | Modified Invariants | INV |
- For other events in the initial model

<table>
<thead>
<tr>
<th>Axioms of the constants</th>
<th>INV</th>
</tr>
</thead>
<tbody>
<tr>
<td>Invariants</td>
<td></td>
</tr>
<tr>
<td>Guard of the event</td>
<td></td>
</tr>
<tr>
<td></td>
<td>INV</td>
</tr>
<tr>
<td>Modified Invariants</td>
<td></td>
</tr>
</tbody>
</table>
- This rule is not mandatory

<table>
<thead>
<tr>
<th>Axiom of the constant Invariants</th>
<th>DLF</th>
</tr>
</thead>
<tbody>
<tr>
<td>⇒ Disjunction of the guards</td>
<td></td>
</tr>
</tbody>
</table>
Refinement Rules (1): Guard Strengthening

- For old events only

<table>
<thead>
<tr>
<th>Axioms of the constants</th>
<th>GRD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstract invariants</td>
<td></td>
</tr>
<tr>
<td>Concrete invariants</td>
<td></td>
</tr>
<tr>
<td>Concrete guards</td>
<td></td>
</tr>
<tr>
<td>⇒</td>
<td></td>
</tr>
<tr>
<td>Abstract guards</td>
<td></td>
</tr>
</tbody>
</table>
- For init event only

<table>
<thead>
<tr>
<th>Axioms of the constants</th>
<th>INV</th>
</tr>
</thead>
<tbody>
<tr>
<td>⇒</td>
<td></td>
</tr>
<tr>
<td>Modified concrete invariants</td>
<td></td>
</tr>
</tbody>
</table>
Refinement Rules (3): Invariant Preservation

- For all events (except init)

- New events refine an implicit non-guarded event with skip action

<table>
<thead>
<tr>
<th>Axioms of the constants</th>
<th>INV</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstract invariant</td>
<td></td>
</tr>
<tr>
<td>Concrete invariant</td>
<td></td>
</tr>
<tr>
<td>Concrete guard</td>
<td></td>
</tr>
<tr>
<td>⇒</td>
<td></td>
</tr>
<tr>
<td>Modified concrete invariant</td>
<td></td>
</tr>
</tbody>
</table>
- For new events only

<table>
<thead>
<tr>
<th>Axioms of the constants</th>
<th>Variant $\in \mathbb{N}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstract invariants</td>
<td></td>
</tr>
<tr>
<td>Concrete invariants</td>
<td></td>
</tr>
<tr>
<td>Concrete guard of a new event</td>
<td></td>
</tr>
</tbody>
</table>

NAT
- For new events only

<table>
<thead>
<tr>
<th>Axioms of the constants</th>
<th>VAR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstract invariants</td>
<td></td>
</tr>
<tr>
<td>Concrete invariants</td>
<td></td>
</tr>
<tr>
<td>Disj. of abs. guards</td>
<td></td>
</tr>
<tr>
<td>⇒</td>
<td></td>
</tr>
<tr>
<td>Disj. of conc. guards</td>
<td></td>
</tr>
</tbody>
</table>
- Global proof rule

<table>
<thead>
<tr>
<th>Axioms of the constants</th>
<th>DLF</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstract invariants</td>
<td></td>
</tr>
<tr>
<td>Concrete invariants</td>
<td></td>
</tr>
<tr>
<td>Disjunction of abstract guards</td>
<td></td>
</tr>
<tr>
<td>⇒</td>
<td></td>
</tr>
<tr>
<td>Disjunction of concrete guards</td>
<td></td>
</tr>
</tbody>
</table>
- For old events (in case of superposition)

<table>
<thead>
<tr>
<th>Axioms of constants</th>
<th>SIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstract invariants</td>
<td></td>
</tr>
<tr>
<td>Concrete invariants</td>
<td></td>
</tr>
<tr>
<td>Concrete guards</td>
<td></td>
</tr>
</tbody>
</table>

⇒
Same actions on common variables