Event-B Course

2. Controlling Cars on a Bridge

Jean-Raymond Abrial

September-October-November 2011

Purpose of this Lecture (1)

- 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

- With each refinement, the observer "zooms in" to see more details

Purpose of this Lecture (2)

- 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 criteria are formulated as proof obligations
- Proofs will be performed by using the sequent calculus
- Inference rules used in the sequent calculus will be reviewed

What you will Learn

- The concepts of state and events for defining models

- Some principles of system development: invariants and refinement

- A refresher of classical logic and simple arithmetic foundations

- A refresher of formal proofs

Outline

1. Presentation of the requirement document

2. Defining the refinement strategy

3. Development of the initial model and the refinements

Remark: Theoretical background provided during development

1. The Requirements Document

Requirement Document (1)

- We show the embedding of the Explanations and the References

- Explanation:

- 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.

Kinds of Requirements (labels)

- There are two kinds of requirements:

    - the equipment (environment) labeled EQP,

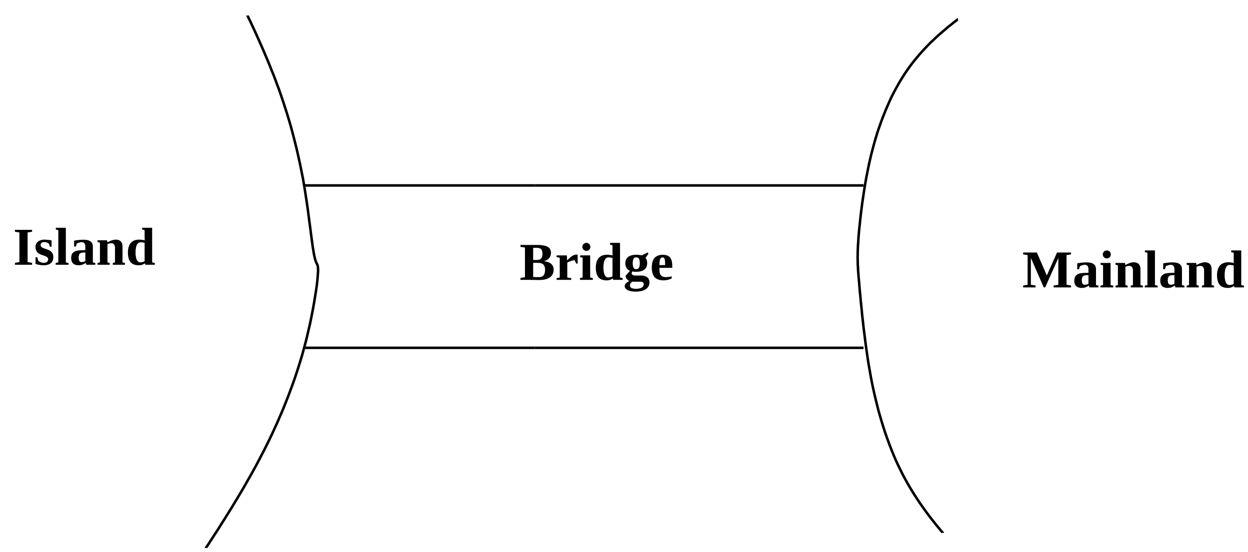
    - the function of the system, labeled FUN.

Requirement Document (2)

- Reference:

The system is controlling cars on a bridge between the mainland and an island	FUN-1

- Explanation: This can be illustrated as follows



Requirements Document (3)

- Explanation: The controller is equipped with two traffic lights.

- Reference:

The system has two traffic lights with two colors: green and red	EQP-1

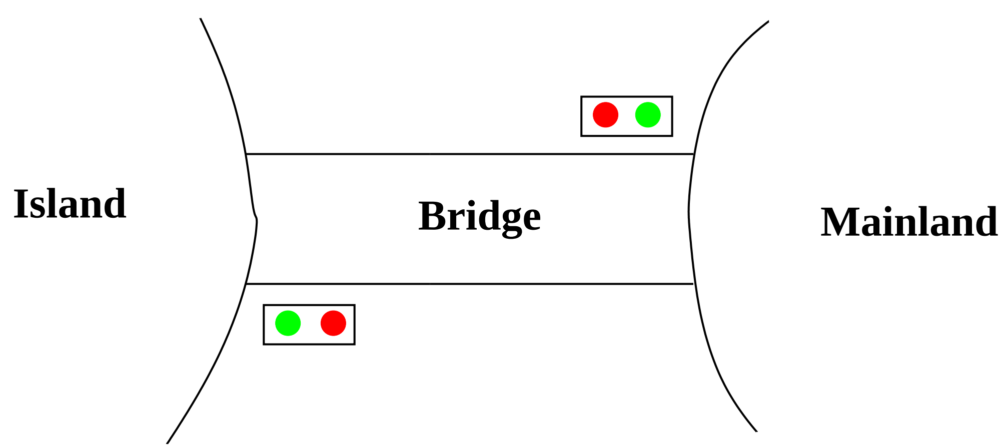
Requirements Document (4)

- Explanation:

- One of the traffic lights is situated on the mainland.

- The other one on the island.

- This can be illustrated as follows:



Requirements Document (5)

- Reference:

The traffic lights control the entrance to the bridge at both ends of it	EQP-2

- Explanation: Drivers are supposed to obey the traffic light

- Reference:

Cars are not supposed to pass on a red traffic light, only on a green one	EQP-3

Requirements Document (6)

- Explanation:

- There are also four car sensors

- These sensors are situated at both ends of the bridge.

- They are supposed to detect the presence of cars

- Reference:

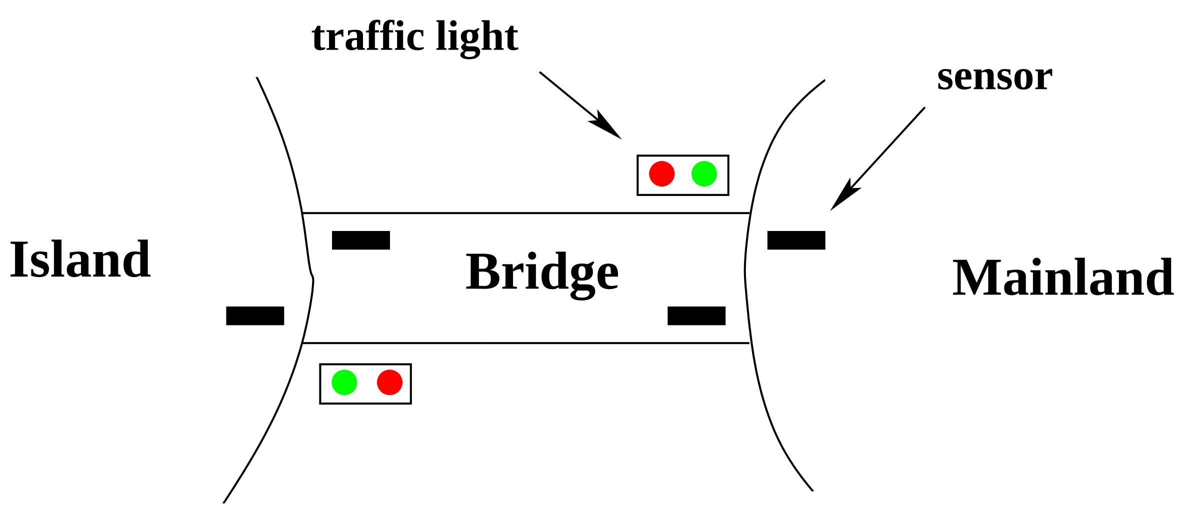
The system is equipped with four car sensors each with two states: on or off	EQP-4

Requirements Document (8)

- Reference:

The sensors are used to detect the presence of cars entering or leaving the bridge	EQP-5

- Explanation: The pieces of equipment can be illustrated as follows:



Requirements Document (9)

- Explanation: This system has two main constraints:

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

    - the bridge is one way.

Requirements Document (10)

- Reference:

The number of cars on the bridge and the island is limited	FUN-2

The bridge is one way or the other, not both at the same time	FUN-3

Summary of the References (1)

The system is controlling cars on a bridge between the mainland and an island	FUN-1

The number of cars on the bridge and the island is limited	FUN-2

The bridge is one way or the other, not both at the same time	FUN-3

Summary of the References (2)

The system has two traffic lights with two colors: green and red	EQP-1

The traffic lights control the entrance to the bridge at both ends of it	EQP-2

Cars are not supposed to pass on a red traffic light, only on a green one	EQP-3

Summary of the References (3)

The system is equipped with four car sensors each with two states: on or off	EQP-4

The sensors are used to detect the presence of cars entering or leaving the bridge	EQP-5

2. The Refinement Strategy

Role of the Refinement Strategy

- Defining a priori the various refinement steps

- Linking these steps with the requirements

- Goal: starting the traceability of the requirements

- Might be modified during the formal development

Our Refinement Strategy (1)

- Initial model: Limiting the number of cars (FUN-2)

The number of cars on the bridge and the island is limited	FUN-2

- First refinement: Introducing the one way bridge (FUN-3)

The bridge is one way or the other, not both at the same time	FUN-3

Our Refinement Strategy (2)

- Second refinement: Introducing the traffic lights (EQP-1,2,3)

- Third refinement: Introducing the sensors (EQP-4,5)

3. The Formal Development

Our Refinement Strategy

- 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)

Initial Model

- It is very simple

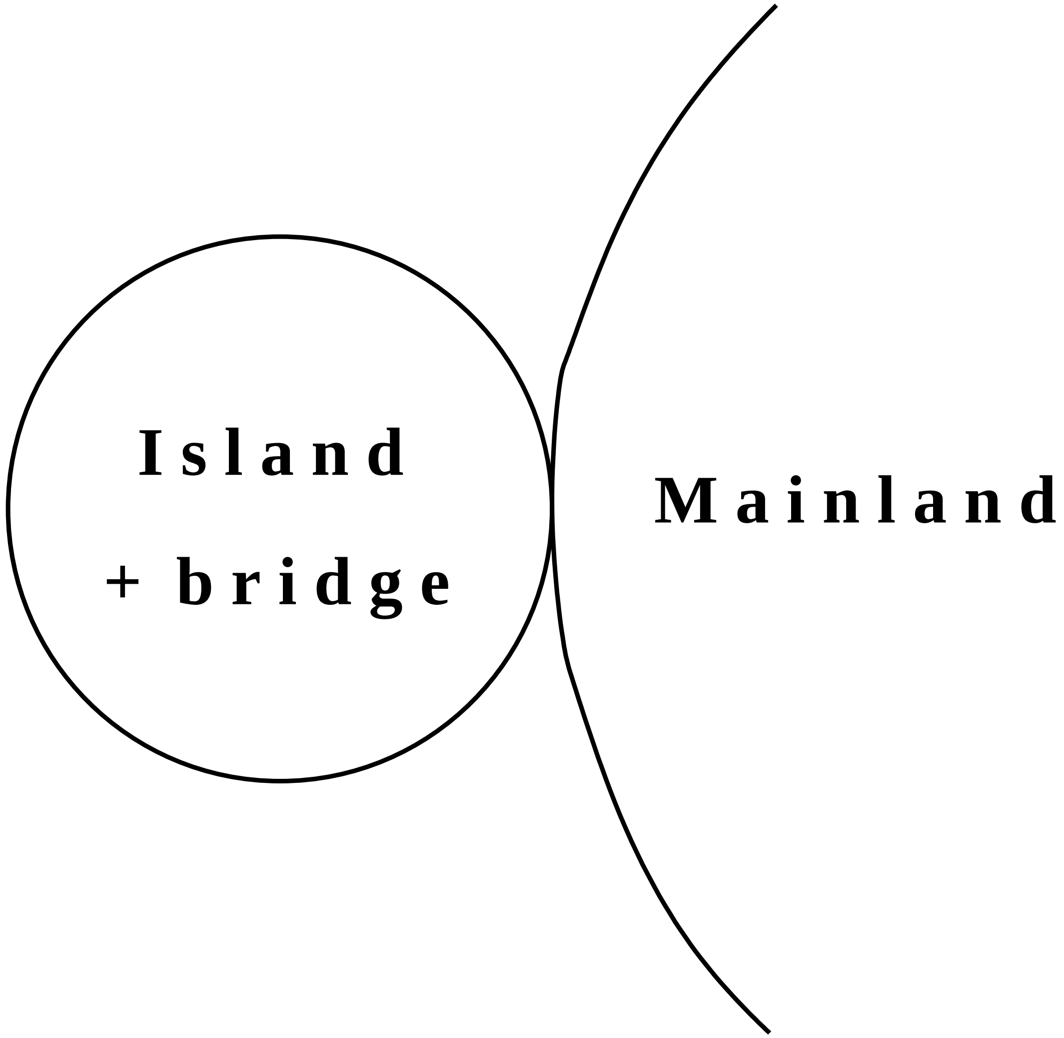
- We completely ignore the equipment: traffic lights and sensors

- We do not even consider the bridge

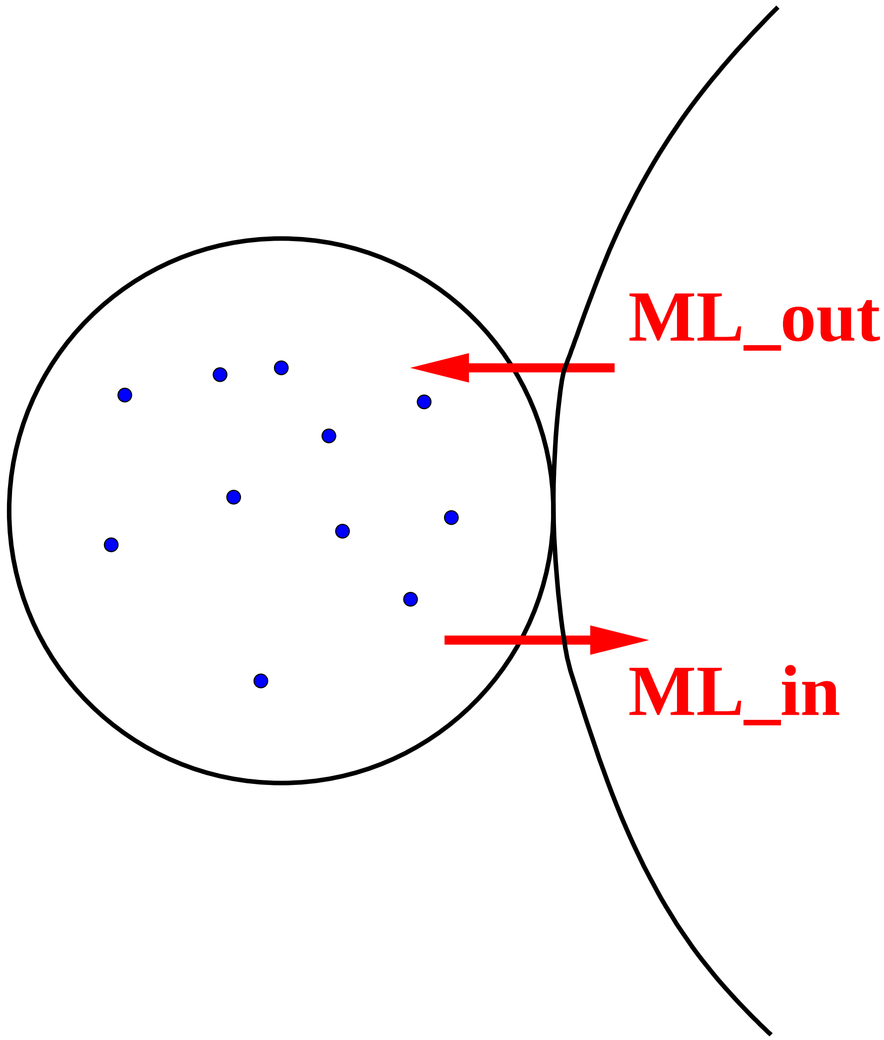
- We are just interested in the pair "island-bridge"

- We are focusing FUN-2: limited number of cars on island-bridge

A Situation as Seen from the Sky



Two Events that may be Observed



Formalizing the State: Constants and Axioms

- STATIC PART of the state: constant d with axiom **axm0_1**

constant: d	axm0_1: $d \in \mathbb{N}$

- d is the maximum number of cars allowed on the Island-Bridge

- **axm0_1** states that d is a natural number

- Constant d is a member of the set $\mathbb{N} = \{0, 1, 2, , \ldots\}$

Formalizing the State: variable

- DYNAMIC PART: variable n with invariants **inv0_1** and **inv0_2**

variable: n

inv0_1: $n \in \mathbb{N}$

inv0_2: $n \leq d$

- n is the effective number of cars on the Island-Bridge

- n is a natural number (**inv0_1**)

- n is always smaller than or equal to d (**inv0_2**): this is FUN_2

Naming Conventions

- Labels **axm0_1**, **inv0_1**, ... are chosen systematically

- The label **axm** or **inv** recalls the purpose:
  axiom of constants or invariant of variables

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

- Later we will have **inv1_1** for an invariant of refinement **1**, etc.

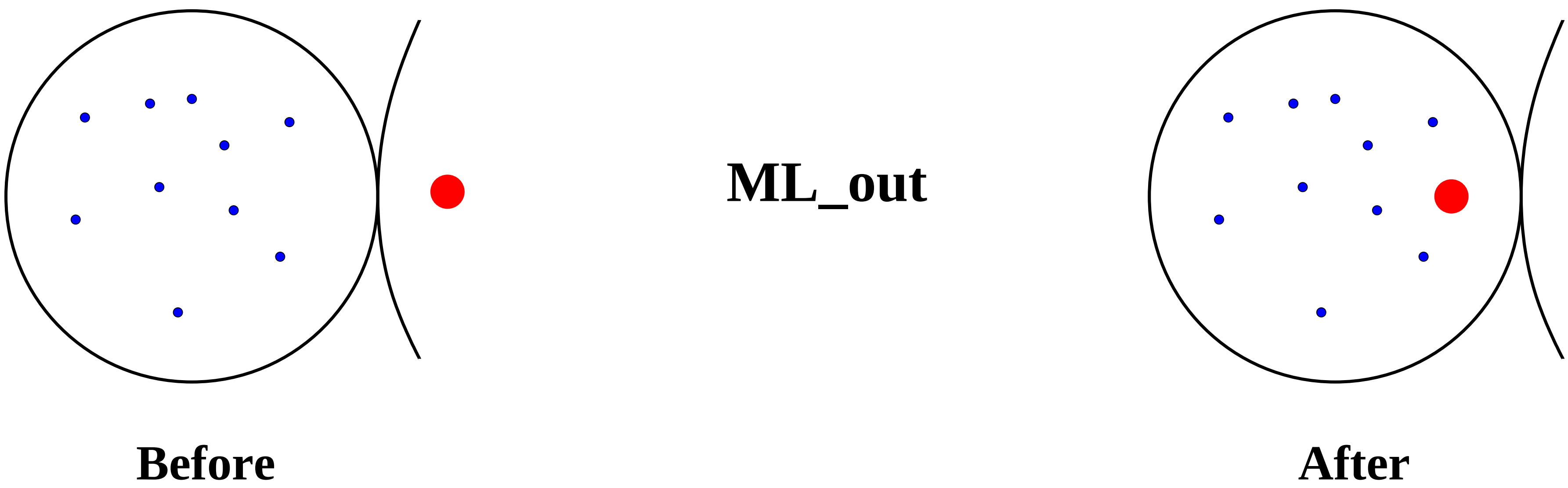
- The **1** like in **inv0_1** is a serial number

- Any convention is valid as long as it is systematic

Event ML_out

- This is the first transition (or event) that can be observed

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

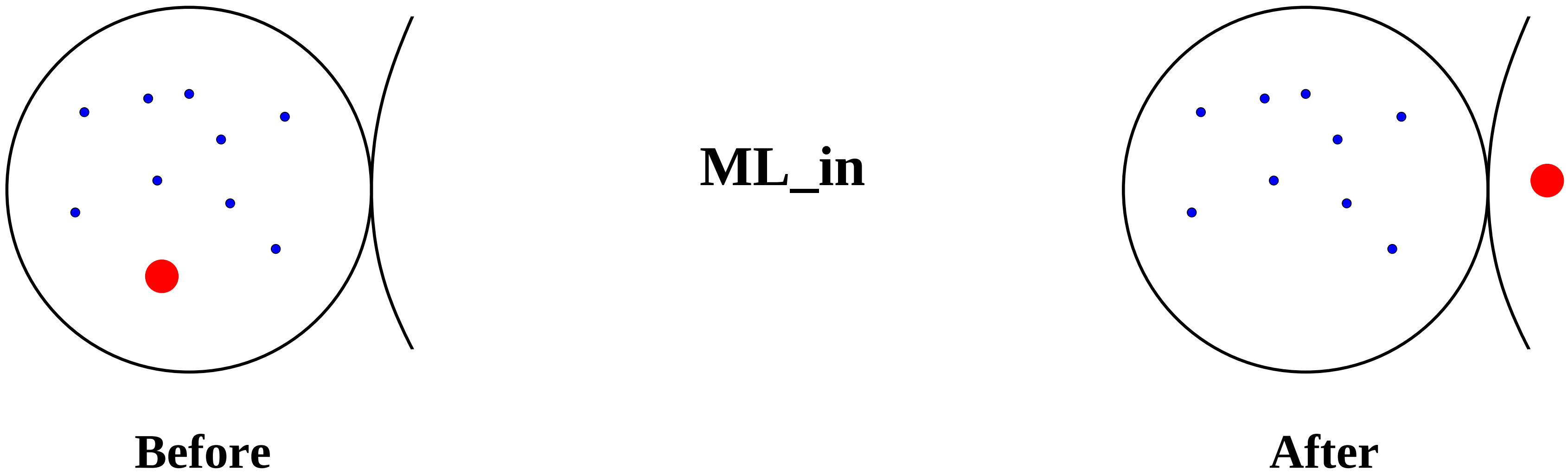


- The number of cars in the Island-Bridge is incremented

Event ML_in

- 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 is decremented

Formalizing the two Events: an Approximation

- Event ML_out increments the number of cars

ML_out

$$n := n + 1$$

- Event ML_in decrements the number of cars

ML_in

$$n := n - 1$$

- An event is denoted by its name and its action (an assignment)

Why an Approximation?

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 have to perform a proof in order to verify this consistency.

Invariants

- An invariant is a constraint on the allowed values of the variables

- An invariant must hold on all reachable states of a model

- To verify that this holds we must show that

    1. the invariant holds for initial states (later), and

    2. the invariant is preserved by all events (following slides)

- We will formalize these two statements as proof obligations (POs)

- We need a rigorous proof showing that these POs indeed hold

Towards the Proof: Before-after Predicates

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

- It describes the relation between the values of the variable(s) *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 *primed* variable name, say n'

Before-after Predicate Examples

The Events

ML_out
$n := n + 1$

ML_in
$n := n - 1$

The corresponding before-after predicates

$$n' = n + 1$$

$$n' = n - 1$$

These representations are equivalent.

About the Shape of the Before-after Predicates

- The before-after predicates we have shown are very simple

$$n' = n + 1 \qquad\qquad n' = n - 1$$

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

- This is because the corresponding events are deterministic

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

$$n' \in \{n + 1, n + 2\}$$

Intuition about Invariant Preservation

- Let us consider invariant **inv0_1**

$$n \in \mathbb{N}$$

- And let us consider event ML_out with before-after predicate

$$n' = n + 1$$

- Preservation of **inv0_1** means that we have (just after ML_out):

$$n' \in \mathbb{N} \quad \text{that is} \quad n + 1 \in \mathbb{N}$$

Being more Precise

- Under hypothesis $n \in \mathbb{N}$ the conclusion $n + 1 \in \mathbb{N}$ holds

- This can be written as follows

$$n \in \mathbb{N} \quad \vdash \quad n + 1 \in \mathbb{N}$$

- This type of statement is called a sequent (next slide)

- Sequent above: invariant preservation proof obligation for **inv0_1**

- More General form of this PO will be introduced shortly

Sequents

- A sequent is a formal statement of the following shape

$$\mathbf{H} \;\vdash\; \mathbf{G}$$

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

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

- The symbol "$\vdash$", called the turnstyle, stands for provability.

  It is read: "Assumptions **H** yield conclusion **G**"

Proof Obligation: Invariant Preservation (1)

- We collectively denote our set of constants by c

- We denote our set of axiomss 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 $I(c, v)$: $I_1(c, v), I_2(c, v), \ldots$

Proof Obligation: Invariant Preservation (2)

- We are given an event with before-after predicate $v' = E(c, v)$

- The following sequent expresses preservation of invariant $I_i(c, v)$:

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

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

- We have given the name INV to this proof obligation

Explanation of the Proof Obligation

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

- We assume that $A(c)$ as well as $I(c,v)$ hold just before the occurrence of the event represented by $v' = E(c,v)$

- Just after the occurrence, invariant $I_i(c,v)$ becomes $I_i(c,v')$, that is, $I_i(c, E(c,v))$

- The predicate $I_i(c, E(c,v))$ must then hold for $I_i(c,v)$ to be an invariant

Vertical Layout of Proof Obligations

- The proof obligation

$\boldsymbol{A(c),\ I(c,v)\ \vdash\ I_i(c,E(c,v))}$	INV

can be re-written vertically as follows:

Axioms Invariants $\vdash$ Modified Invariant

$\boldsymbol{A(c)}$ $\boldsymbol{I(c,v)}$ $\vdash$ $\boldsymbol{I_i(c,E(c,v))}$	INV

Back to our Example

- We have two events

ML_out
$n := n + 1$

ML_in
$n := n - 1$

- And two invariants

inv0_1: $n \in \mathbb{N}$

inv0_2: $n \leq d$

- Thus, we need to prove four proof obligations

Proof obligation for ML_out and inv0_1

ML_out

$n := n + 1$

$(n' = n + 1)$

Axiom **axm0_1**
Invariant **inv0_1**
Invariant **inv0_2**
$\vdash$
Modified Invariant **inv0_1**

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

- This proof obligation is named: ML_out / **inv0_1** / INV

Proof obligation for ML_out and inv0_2

ML_out
$n := n + 1$

$(n' = n + 1)$

Axiom **axm0_1**
Invariant **inv0_1**
Invariant **inv0_2**
$\vdash$
Modified Invariant **inv0_2**

$$
\begin{array}{l}
d \in \mathbb{N} \\
n \in \mathbb{N} \\
n \leq d \\
\vdash \\
n + 1 \leq d
\end{array}
$$

- This proof obligation is named: ML_out / **inv0_2** / INV

Proof obligation for ML_in and inv0_1

ML_in

$n := n - 1$

$(n' = n - 1)$

Axiom **axm0_1**
Invariant **inv0_1**
Invariant **inv0_2**
$\vdash$
Modified Invariant **inv0_1**

$d \in \mathbb{N}$
$n \in \mathbb{N}$
$n \leq d$
$\vdash$
$n - 1 \in \mathbb{N}$

- This proof obligation is named: ML_in / **inv0_1** / INV

Proof obligation for ML_in and inv0_2

ML_in

$n := n - 1$

$(n' = n - 1)$

Axiom **axm0_1**
Invariant **inv0_1**
Invariant **inv0_2**
$\vdash$
Modified Invariant **inv0_2**

$$d \in \mathbb{N}$$
$$n \in \mathbb{N}$$
$$n \leq d$$
$$\vdash$$
$$n - 1 \leq d$$

- This proof obligation is named: ML_in / **inv0_2** / INV

Summary of Proof Obligations

ML_out / **inv0_1** / INV

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

ML_out / **inv0_2** / INV

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

ML_in / **inv0_1** / INV

$$d \in \mathbb{N}$$
$$n \in \mathbb{N}$$
$$n \leq d$$
$$\vdash$$
$$n - 1 \in \mathbb{N}$$

ML_in / **inv0_2** / INV

$$d \in \mathbb{N}$$
$$n \in \mathbb{N}$$
$$n \leq d$$
$$\vdash$$
$$n - 1 \leq d$$

Informal Proof of ML_out / inv0_1 / INV

$$\begin{array}{l} \quad d \in \mathbb{N} \\ \quad n \in \mathbb{N} \\ \quad n \leq d \\ \vdash \\ \quad n+1 \in \mathbb{N} \end{array} \quad \underset{\text{hypotheses}}{\overset{\text{remove}}{\longrightarrow}} \quad \begin{array}{l} \quad n \in \mathbb{N} \\ \vdash \\ \quad n+1 \in \mathbb{N} \end{array} \quad \text{obvious}$$

- In the first step, we remove some irrelevant hypotheses

- In the second and final step, we accept the sequent as it is

- We have implicitly applied inference rules

- For rigorous reasoning we will make these rules explicit

Inference Rules

$$\frac{\mathsf{H}_1 \ \vdash \ \mathsf{G}_1 \qquad \cdots \qquad \mathsf{H}_n \ \vdash \ \mathsf{G}_n}{\mathsf{H} \ \vdash \ \mathsf{G}} \ \textsf{RULE\_NAME}$$

- Above horizontal line: n sequents called antecedents ($n \geq 0$)

- Below horizontal line: exactly one sequent called consequent

- To prove the consequent, it is sufficient to prove the antecedents

- A rule with no antecedent ($n = 0$) is called an axiom

Inference Rule: Monotonicity of Hypotheses

- The rule that removes hypotheses can be stated as follows:

$$\frac{\mathbf{H} \;\vdash\; \mathbf{G}}{\mathbf{H, H'} \;\vdash\; \mathbf{G}} \quad \mathbf{MON}$$

- It expresses the monotonicity of the hypotheses

Some Arithmetic Rules of Inference

- The Second Peano Axiom

$$\frac{}{\mathbf{n} \in \mathbb{N} \;\vdash\; \mathbf{n} + 1 \in \mathbb{N}} \quad \textbf{P2}$$

$$\frac{}{0 < \mathbf{n} \;\vdash\; \mathbf{n} - 1 \in \mathbb{N}} \quad \textbf{P2}'$$

More Arithmetic Rules of Inference

- Axioms about ordering relations on the integers

$$\frac{}{\mathbf{n} < \mathbf{m} \;\vdash\; \mathbf{n} + 1 \leq \mathbf{m}} \quad \textbf{INC}$$

$$\frac{}{\mathbf{n} \leq \mathbf{m} \;\vdash\; \mathbf{n} - 1 \leq \mathbf{m}} \quad \textbf{DEC}$$

Application of Inference Rules

- Consider again the 2nd Peano axiom:

$$\frac{}{\mathbf{n} \in \mathbb{N} \vdash \mathbf{n} + 1 \in \mathbb{N}} \quad \textbf{P2}$$

- It is a rule schema where **n** is called a meta-variable

- It can be applied to following sequent by matching $\boldsymbol{a} + \boldsymbol{b}$ with **n**:

$$\boldsymbol{a} + \boldsymbol{b} \in \mathbb{N} \vdash \boldsymbol{a} + \boldsymbol{b} + \mathbf{1} \in \mathbb{N}$$

Proofs

- A proof is a tree of sequents with axioms at the leaves.

- The rules applied to the leaves are axioms.

- Each sequent is labeled with (name of) proof rule applied to it.

- The sequent at the root of the tree is called the root sequent.

- The purpose of a proof is to establish the truth of its root sequent.

A Formal Proof of: ML_out / inv0_1 / INV

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

MON

$$n \in \mathbb{N}$$
$$\vdash$$
$$n + 1 \in \mathbb{N}$$

P2

- Proof requires only application of two rules: **MON** and **P2**

A Failed Proof Attempt: ML_out / inv0_2 / INV

$$\begin{array}{l} \quad d \in \mathbb{N} \\ \quad n \in \mathbb{N} \\ \quad n \leq d \\ \vdash \\ \quad n+1 \leq d \end{array} \quad \text{MON} \quad \begin{array}{l} \quad n \leq d \\ \vdash \\ \quad n+1 \leq d \end{array} \quad ?$$

- We put a **?** to indicate that we have no rule to apply

- The proof fails: we cannot conclude with rule **INC** ($n < d$ needed)

$$\frac{}{\mathbf{n} < \mathbf{m} \;\vdash\; \mathbf{n}+1 \leq \mathbf{m}} \quad \textbf{INC}$$

A Failed Proof Attempt: ML_in / inv0_1 / INV

$$\begin{array}{l} d \in \mathbb{N} \\ n \in \mathbb{N} \\ n \leq d \\ \vdash \\ n - 1 \in \mathbb{N} \end{array} \quad \text{MON} \quad \begin{array}{l} n \in \mathbb{N} \\ \vdash \\ n - 1 \in \mathbb{N} \end{array} \quad ?$$

- The proof fails: we cannot conclude with rule **P2′** ($0 < n$ needed)

$$\frac{}{0 < \mathbf{n} \vdash \mathbf{n} - 1 \in \mathbb{N}} \quad \mathbf{P2'}$$

A Formal Proof of: ML_in / inv0_2 / INV

$$\begin{array}{l} \quad d \in \mathbb{N} \\ \quad n \in \mathbb{N} \\ \quad n \leq d \\ \vdash \\ \quad n - 1 \leq d \end{array}$$

MON

$$\begin{array}{l} \quad n \leq d \\ \vdash \\ \quad n - 1 \leq d \end{array}$$

DEC

$$\frac{}{\mathbf{n} \leq \mathbf{m} \vdash \mathbf{n} - 1 \leq \mathbf{m}} \quad \textbf{DEC}$$

Reasons for Proof Failure

- We needed hypothesis $n < d$ to prove ML_out / **inv0_2** / INV

- We needed hypothesis $0 < n$ to prove ML_in / **inv0_1** / INV

ML_out $n := n + 1$	**ML_in** $n := n - 1$

- We are going to add $n < d$ as a guard to event ML_out

- We are going to add $0 < n$ as a guard to event ML_in

Improving the Events: Introducing Guards

ML_out
  when
    $n < d$
  then
    $n := n + 1$
  end

ML_in
  when
    $0 < n$
  then
    $n := n - 1$
  end

- We are adding guards to the events

- The guard is the necessary condition for an event to “occur”

Proof Obligation: General Invariant Preservation

- 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 PO as follows:

Axioms Invariants Guard of the event $\vdash$ Modified Invariant		$A(c)$ $I(c,v)$ $G(c,v)$ $\vdash$ $I_i(c, E(c,v))$	INV

A Formal Proof of: ML_out / inv0_1 / INV

$$\begin{array}{l} d \in \mathbb{N} \\ n \in \mathbb{N} \\ n \leq d \\ n < d \\ \vdash \\ n+1 \in \mathbb{N} \end{array} \quad \text{MON} \quad \begin{array}{l} n \in \mathbb{N} \\ \vdash \\ n+1 \in \mathbb{N} \end{array} \quad \text{P2}$$

- Adding new assumptions to a sequent does not affect its provability

$$\begin{array}{l} d \in \mathbb{N} \\ n \in \mathbb{N} \\ n \leq d \\ n < d \\ \vdash \\ n + 1 \leq d \end{array}$$

MON

$$\begin{array}{l} n < d \\ \vdash \\ n + 1 \leq d \end{array}$$

INC

- Now we can conclude the proof using rule **INC**

$$\frac{}{\mathbf{n} < \mathbf{m} \vdash \mathbf{n} + 1 \leq \mathbf{m}} \quad \textbf{INC}$$

A Formal Proof of: ML_in / inv0_1 / INV

$$\begin{array}{l} d \in \mathbb{N} \\ n \in \mathbb{N} \\ n \leq d \\ 0 < n \\ \vdash \\ n - 1 \in \mathbb{N} \end{array} \quad \text{MON} \quad \begin{array}{l} 0 < n \\ \vdash \\ n - 1 \in \mathbb{N} \end{array} \quad \text{P2'}$$

- Now we can conclude the proof using rule **P2′**

$$\frac{}{0 < \mathbf{n} \;\vdash\; \mathbf{n} - 1 \in \mathbb{N}} \quad \textbf{P2}'$$

$$\begin{array}{l} d \in \mathbb{N} \\ n \in \mathbb{N} \\ n \leq d \\ n < d \\ \vdash \\ n - 1 \leq d \end{array}$$

MON

$$\begin{array}{l} n \leq d \\ \vdash \\ n - 1 \leq d \end{array}$$

DEC

- Again, the proof still works after the addition of a new assumption

Re-proving the Events: No Proofs Fail

$$\begin{array}{l} d \in \mathbb{N} \\ n \in \mathbb{N} \\ n \leq d \\ n < d \\ \vdash \\ n+1 \in \mathbb{N} \end{array}$$

$$\begin{array}{l} d \in \mathbb{N} \\ n \in \mathbb{N} \\ n \leq d \\ n < d \\ \vdash \\ n+1 \leq d \end{array}$$

$$\begin{array}{l} d \in \mathbb{N} \\ n \in \mathbb{N} \\ n \leq d \\ 0 < n \\ \vdash \\ n-1 \in \mathbb{N} \end{array}$$

$$\begin{array}{l} d \in \mathbb{N} \\ n \in \mathbb{N} \\ n \leq d \\ 0 < n \\ \vdash \\ n-1 \leq d \end{array}$$

Initialization

- Our system must be initialized (with no car in the island-bridge)

- The initialization event is never guarded

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

-Its before-after predicate is just an after predicate

init
$n := 0$

After predicate

$n' = 0$

Proof Obligation: Invariant Establishment

- 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 PO is the following:

Axioms $\vdash$ Modified Invariant	$A(c)$ $\vdash$ $I_i(c, K(c))$	INV

Applying the Invariant Establishment PO

axm0_1
$\vdash$
Modified **inv0_1**

$$d \in \mathbb{N} \vdash 0 \in \mathbb{N}$$

inv0_1 / INV

axm0_1
$\vdash$
Modified **inv0_2**

$$d \in \mathbb{N} \vdash 0 \leq d$$

inv0_2 / INV

More Arithmetic Inference Rules

- First Peano Axiom

$$\frac{}{\vdash \ \mathbf{0} \in \mathbb{N}} \quad \textbf{P1}$$

- Third Peano Axiom (slightly modified)

$$\frac{}{\mathsf{n} \in \mathbb{N} \ \vdash \ \mathbf{0} \leq \mathsf{n}} \quad \textbf{P3}$$

Proofs of Invariant Establishment

$$d \in \mathbb{N} \vdash 0 \in \mathbb{N}$$

MON

$$\vdash 0 \in \mathbb{N}$$

P1

$$d \in \mathbb{N} \vdash 0 \leq d$$

P3

A Missing Requirement

- 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

Proof Obligation: Deadlock Freedom

- 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:

$A(c)$ $I(c, v)$ $\vdash$ $G_1(c, v) \ \vee \ \ldots \ \vee \ G_m(c, v)$	DLF

Applying the Deadlock Freedom PO

axm0_1 **inv0_1** **inv0_2** $\vdash$ Disjunction of guards	$d \in \mathbb{N}$ $n \in \mathbb{N}$ $n \leq d$ $\vdash$ $n < d \lor 0 < n$

- This cannot be proved with the inference rules we have so far

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

- We continue our proof by a case analysis:
  - case 1: $n = d$
  - case 2: $n < d$

Inference Rules for Disjunction

- Proof by case analysis

$$\frac{\mathbf{H}, \mathbf{P} \ \vdash \ \mathbf{R} \qquad \mathbf{H}, \mathbf{Q} \ \vdash \ \mathbf{R}}{\mathbf{H}, \ \mathbf{P} \lor \mathbf{Q} \ \vdash \ \mathbf{R}} \ \text{OR\_L}$$

- Choice for proving a disjunctive goal

$$\frac{\mathbf{H} \ \vdash \ \mathbf{P}}{\mathbf{H} \ \vdash \ \mathbf{P} \lor \mathbf{Q}} \ \text{OR\_R1}$$

$$\frac{\mathbf{H} \ \vdash \ \mathbf{Q}}{\mathbf{H} \ \vdash \ \mathbf{P} \lor \mathbf{Q}} \ \text{OR\_R2}$$

Proof of Deadlock Freedom

$$\begin{array}{l} d \in \mathbb{N} \\ n \in \mathbb{N} \\ n \leq d \\ \vdash \\ n < d \lor 0 < n \end{array}$$

MON

$$\begin{array}{l} n \leq d \\ \vdash \\ n < d \lor 0 < n \end{array}$$

...

Proof of Deadlock Freedom (cont'd)

$$n \le d \vdash n < d \;\vee\; 0 < n$$

OR_L

$$\left\{ \begin{array}{l} n < d \vdash n < d \;\vee\; 0 < n \quad \ldots \\ \\ n = d \vdash n < d \;\vee\; 0 < n \quad \ldots \end{array} \right.$$

Proof of Deadlock Freedom (cont'd)

$$\left\{ \begin{array}{l} \boxed{\begin{array}{l} n < d \\ \vdash \\ n < d \;\vee\; 0 < n \end{array}} \quad \text{OR\_R1} \quad \boxed{n < d \;\vdash\; n < d} \; ? \\[2em] \boxed{\begin{array}{l} n = d \\ \vdash \\ n < d \;\vee\; 0 < n \end{array}} \; ? \end{array} \right.$$

- The first **?** seems to be obvious

- The second **?** can be (partially) solved by applying the equality

More Inference Rules: Identity and Equality

- The identity axiom (conclusion holds by hypothesis)

$$\frac{}{\mathbf{P} \ \vdash \ \mathbf{P}} \ \mathbf{HYP}$$

- Rewriting an equality (**EQ_LR**) and reflexivity of equality (**EQL**)

$$\frac{\mathbf{H(F)},\ \mathbf{E} = \mathbf{F} \ \vdash \ \mathbf{P(F)}}{\mathbf{H(E)},\ \mathbf{E} = \mathbf{F} \ \vdash \ \mathbf{P(E)}} \ \mathbf{EQ\_LR}$$

$$\frac{}{\vdash \ \mathbf{E} = \mathbf{E}} \ \mathbf{EQL}$$

Proof of Deadlock Freedom (end)

$$\left\{ \begin{array}{l} \boxed{\begin{array}{l} n < d \\ \vdash \\ n < d \;\vee\; 0 < n \end{array}} \quad \text{OR\_R1} \quad \boxed{n < d \;\vdash\; n < d} \quad \text{HYP} \\ \\ \boxed{\begin{array}{l} n = d \\ \vdash \\ n < d \;\vee\; 0 < n \end{array}} \quad \text{EQ\_LR} \quad \boxed{\vdash\; d < d \;\vee\; 0 < d} \quad \text{OR\_R2} \end{array} \right.$$

$$\text{OR\_R2} \quad \boxed{\vdash\; 0 < d} \quad ?$$

- We still have a problem: d must be positive!

Adding the Forgotten Axiom

- If d is equal to 0, then no car can ever enter the Island-Bridge

axm0_2: $0 < d$

Initial Model: Conclusion

- Thanks to the proofs, we discovered 3 errors

- They were corrected by:
    - adding guards to both events
    - adding an axiom

- The interaction of modeling and proving is an essential element of Formal Methods with Proofs

Proof Obligations for Initial Model

- We have seen three kinds of proof obligations:
  - The Invariant Establishment PO: INV
  - The Invariant Preservation PO: INV
  - The Deadlock Freedom PO (optional): DLF

Proof Obligations for Initial Model (cont'd)

Axioms $\vdash$ Modified Invariant	INV

Axioms Invariants Guard of the event $\vdash$ Modified Invariant	INV

Axiom Invariants $\vdash$ Disjunction of the guards	DLF

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
when
$n < d$
then
$n := n + 1$
end

ML_in
when
$0 < n$
then
$n := n - 1$
end

Our Refinement Strategy

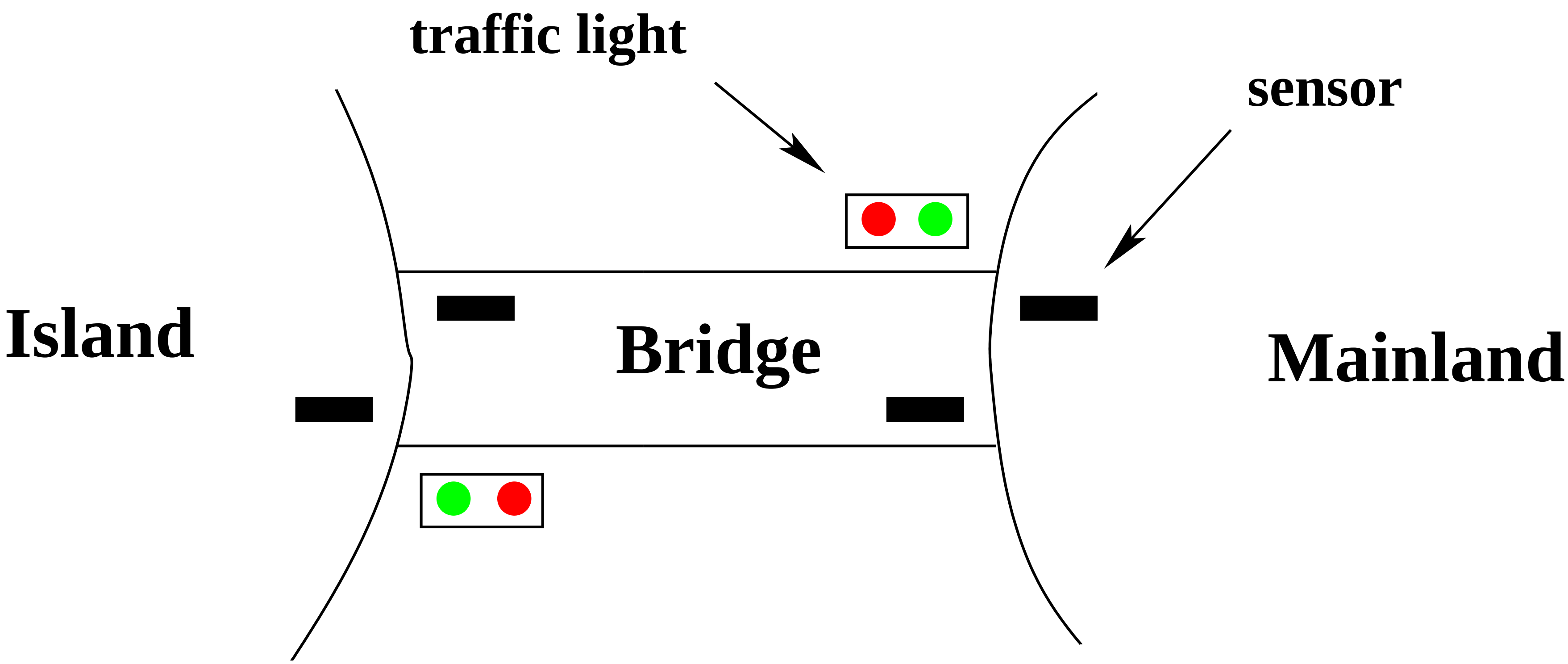
- Initial model: Limiting the number of cars (FUN-2)

- 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



First Refinement: Introducing a One-Way Bridge

- We go down with our parachute

- Our view of the system gets more accurate

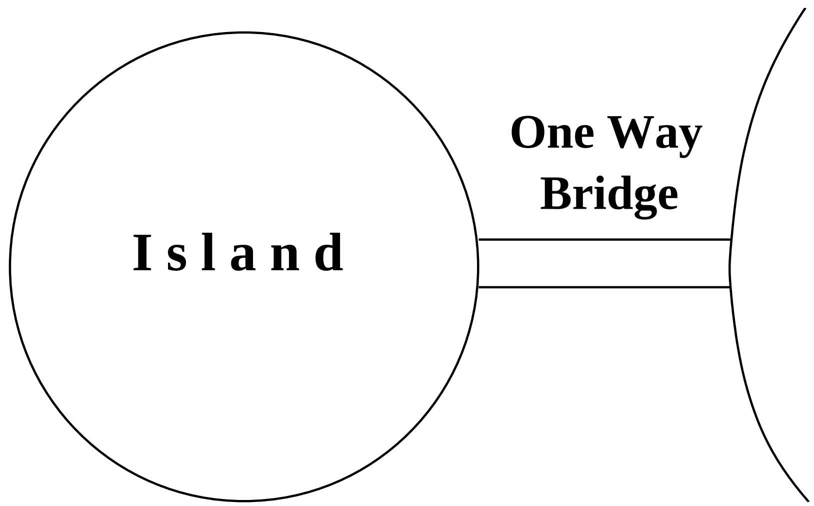
- We introduce the bridge and separate it from the island

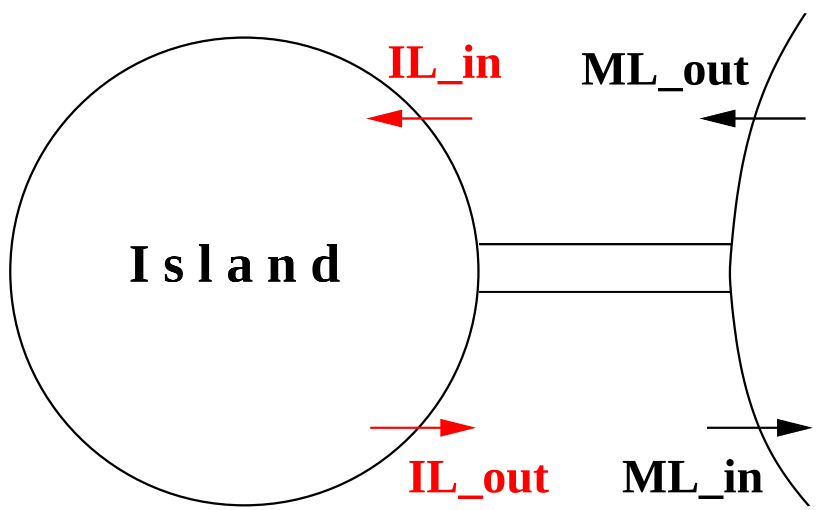
- We refine the state and the events

- We also add two new events: IL_in and IL_out

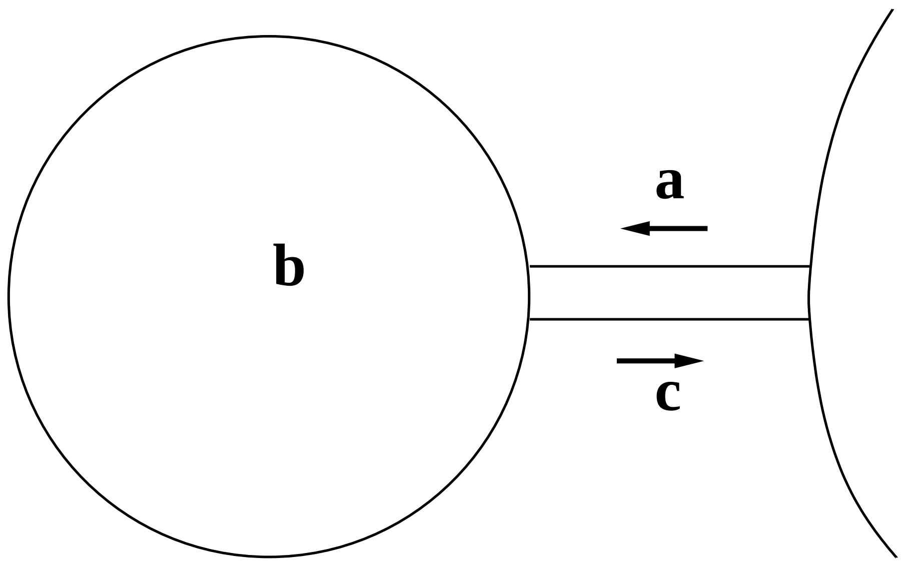
- We are focusing on FUN-3: one-way bridge

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

Refining the State: Formalizing Variables a, b,and c

- Variables $\boldsymbol{a}$, $\boldsymbol{b}$, and $\boldsymbol{c}$ denote natural numbers

$$\boldsymbol{a} \in \mathbb{N}$$

$$\boldsymbol{b} \in \mathbb{N}$$

$$\boldsymbol{c} \in \mathbb{N}$$

Refining the State: Introducing New Invariants

- 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

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

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

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

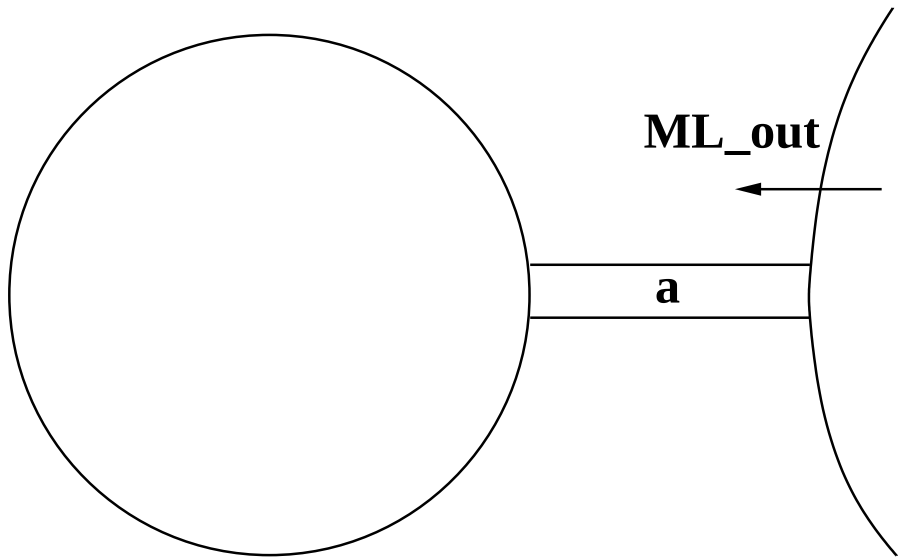
inv1_4: $a + b + c = n$

inv1_5: $a = 0 \quad \vee \quad c = 0$

- Invariants **inv1_1** to **inv1_5** are called the concrete invariants

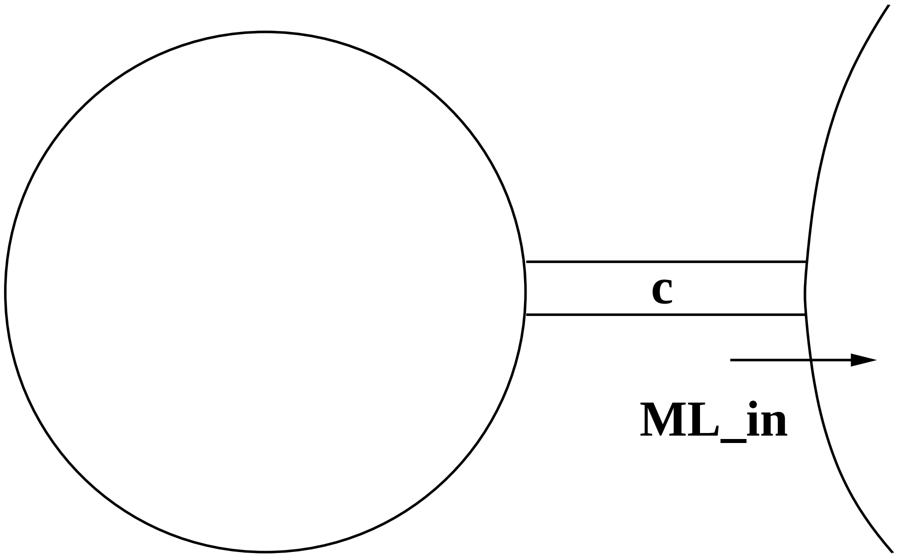
- **inv1_4** glues the abstract state, n, to the concrete state, a, b, c

Proposal for Refining Event ML_out



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

Proposal for Refining Event ML_in



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

B-A Predicates: Preserved Variables

```
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:

$$a' = a + 1 \quad \wedge \quad b' = b \quad \wedge \quad c' = c$$

$$a' = a \quad \wedge \quad b' = b \quad \wedge \quad c' = c - 1$$

Intuition about Refinement

The concrete model behaves as specified by the abstract model (i.e., concrete model does not exhibit any new behaviors)

To show this we have to prove that

1. every concrete event is simulated by its abstract counterpart (event refinement: following slides)
2. to every concrete initial state corresponds an abstract one (initial state refinement: later)

We will make these two conditions more precise and formalize them as proof obligations.

Intuition about refinement (1)


```
(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
```

- The concrete version is not contradictory with the abstract one

- When the concrete version is enabled then so is the abstract one

- Executions seem to be compatible

Intuition about refinement (2)


```
(abstract_)ML_in
  when
    0 < n
  then
    n := n − 1
  end
```

```
(concrete_)ML_in
  when
    0 < c
  then
    c := c − 1
  end
```

- Same remarks as in the previous slide

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

Proof Obligations for Refinement

- The concrete guard is stronger than the abstract one

- Each concrete action is compatible with its abstract counterpart

Proving Correct Refinement: the Situation

Constants c with axioms $A(c)$

Abstract variables v with abstract invariant $I(c, v)$

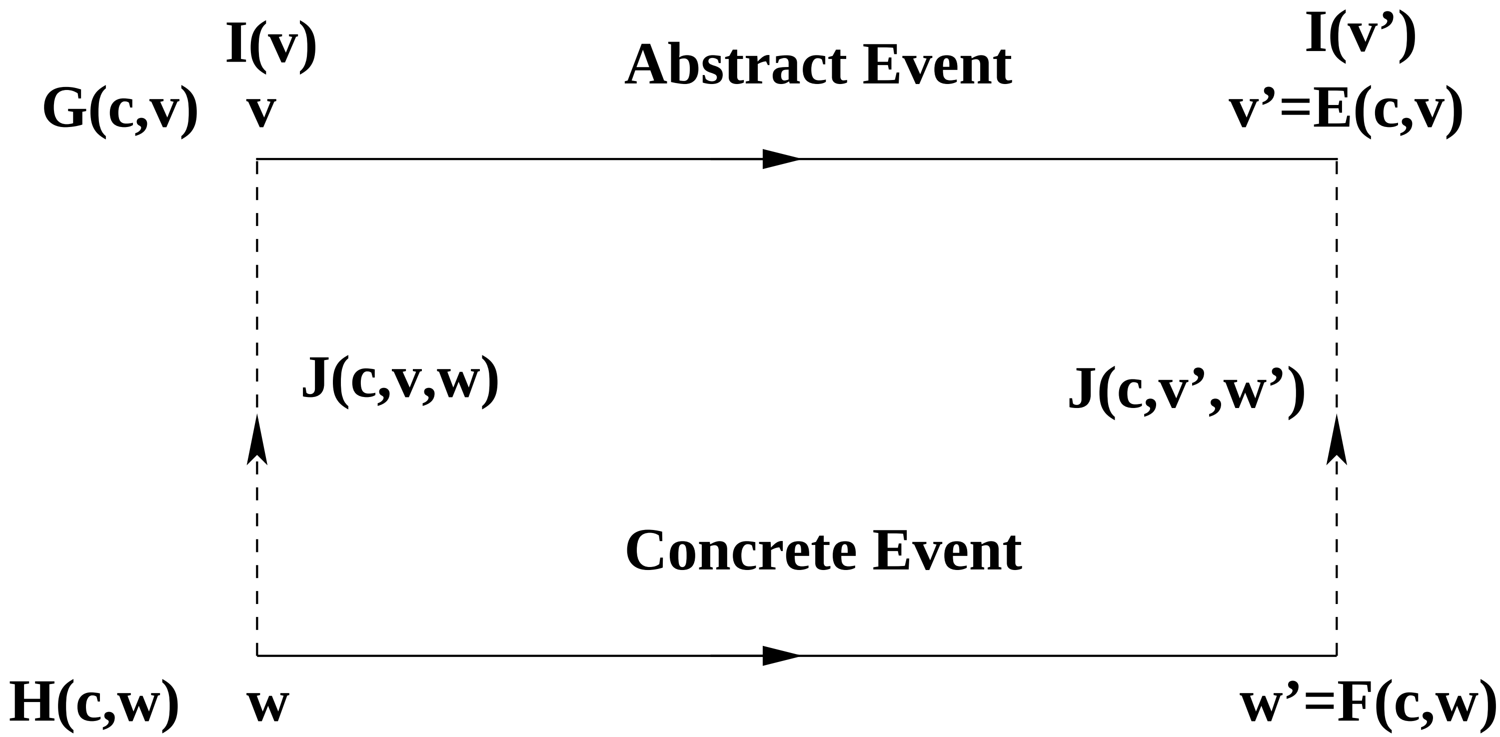
Concrete variables w with concrete invariant $J(c, v, w)$

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

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

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

Correctness of Event Refinement



1. The concrete guard is stronger than the abstract one (Guard Strengthening, following slides)

2. Each concrete action is simulated by its abstract counterpart (Concrete Invariant Preservation, later)

Proof Obligation: Guard Strengthening

Axioms Abstract Invariant Concrete Invariant Concrete Guard $\vdash$ Abstract Guard	$A(c)$ $I(c, v)$ $J(c, v, w)$ $H(c, w)$ $\vdash$ $G_i(c, v)$	GRD

Proof Obligations for Guard Strengthening

- 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

$\vdash$

Abstract guard of ML_out

$$
\begin{array}{l}
d \in \mathbb{N} \\
0 < d \\
n \in \mathbb{N} \\
n \leq d \\
a \in \mathbb{N} \\
b \in \mathbb{N} \\
c \in \mathbb{N} \\
a + b + c = n \\
a = 0 \quad \lor \quad c = 0 \\
a + b < d \\
c = 0 \\
\vdash \\
n < d
\end{array}
$$

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 ML_out / GRD

$$\begin{array}{l} d \in \mathbb{N} \\ 0 < d \\ n \in \mathbb{N} \\ n \leq d \\ a \in \mathbb{N} \\ b \in \mathbb{N} \\ c \in \mathbb{N} \\ a+b+c=n \\ a=0 \;\vee\; c=0 \\ a+b<d \\ c=0 \\ \vdash \\ n<d \end{array} \quad \text{MON} \quad \begin{array}{l} a+b+c=n \\ a+b<d \\ c=0 \\ \vdash \\ n<d \end{array} \quad \text{EQ\_LR} \quad \begin{array}{l} a+b+0=n \\ a+b<d \\ \vdash \\ n<d \end{array} \quad \text{ARITH} \;\ldots$$

$$\ldots \quad \begin{array}{l} a+b=n \\ a+b<d \\ \vdash \\ n<d \end{array} \quad \text{EQ\_LR} \quad \begin{array}{l} n<d \\ \vdash \\ n<d \end{array} \quad \text{HYP}$$

The "rule" name ARITH stands for simple arithmetic simplifications.

Applying Guard Strengthening to Event ML_in

axm0_1	$d \in \mathbb{N}$
axm0_2	$0 < d$
inv0_1	$n \in \mathbb{N}$
inv0_2	$n \leq d$
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	$a = 0 \quad \lor \quad c = 0$
Concrete guard of ML_in	$0 < c$
$\vdash$	$\vdash$
Abstract guard of ML_in	$0 < n$

ML_in / GRD

(abstract-)ML_in
when
$0 < n$
then
$n := n - 1$
end

(concrete-)ML_in
when
$0 < c$
then
$c := c - 1$
end

Proof of ML_in / GRD

$$
\begin{array}{l}
d \in \mathbb{N} \\
0 < d \\
n \in \mathbb{N} \\
n \leq d \\
a \in \mathbb{N} \\
b \in \mathbb{N} \\
c \in \mathbb{N} \\
a + b + c = n \\
a = 0 \;\lor\; c = 0 \\
0 < c \\
\vdash \\
0 < n
\end{array}
$$

MON

$$
\begin{array}{l}
b \in \mathbb{N} \\
a + b + c = n \\
a = 0 \;\lor\; c = 0 \\
0 < c \\
\vdash \\
0 < n
\end{array}
$$

OR_L

$$
\begin{array}{l}
b \in \mathbb{N} \\
a + b + c = n \\
a = 0 \\
0 < c \\
\vdash \\
0 < n
\end{array}
$$

EQ_LR $\ldots$

$$
\begin{array}{l}
b \in \mathbb{N} \\
a + b + c = n \\
c = 0 \\
0 < c \\
\vdash \\
0 < n
\end{array}
$$

EQ_LR $\ldots$

Proof of ML_in / GRD

$$\begin{array}{l} d \in \mathbb{N} \\ 0 < d \\ n \in \mathbb{N} \\ n \leq d \\ a \in \mathbb{N} \\ b \in \mathbb{N} \\ c \in \mathbb{N} \\ a + b + c = n \\ a = 0 \;\lor\; c = 0 \\ 0 < c \\ \vdash \\ 0 < n \end{array}$$

MON

$$\begin{array}{l} b \in \mathbb{N} \\ a + b + c = n \\ a = 0 \;\lor\; c = 0 \\ 0 < c \\ \vdash \\ 0 < n \end{array}$$

OR_L

$$\begin{array}{l} b \in \mathbb{N} \\ a + b + c = n \\ a = 0 \\ 0 < c \\ \vdash \\ 0 < n \end{array}$$

EQ_LR ...

$$\begin{array}{l} b \in \mathbb{N} \\ a + b + c = n \\ c = 0 \\ 0 < c \\ \vdash \\ 0 < n \end{array}$$

EQ_LR ...

...

$$\begin{array}{l} b \in \mathbb{N} \\ 0 + b + c = n \\ 0 < c \\ \vdash \\ 0 < n \end{array}$$

ARITH

$$\begin{array}{l} b \in \mathbb{N} \\ b + c = n \\ 0 < c \\ \vdash \\ 0 < n \end{array}$$

ARITH

$$\begin{array}{l} c \leq n \\ 0 < c \\ \vdash \\ 0 < n \end{array}$$

ARITH

$$\begin{array}{l} 0 < n \\ \vdash \\ 0 < n \end{array}$$

HYP

...

$$\begin{array}{l} b \in \mathbb{N} \\ a + b + 0 = n \\ 0 < 0 \\ \vdash \\ 0 < n \end{array}$$

ARITH

$$\begin{array}{l} b \in \mathbb{N} \\ a + b = n \\ 0 < 0 \\ \vdash \\ 0 < n \end{array}$$

MON

$$\begin{array}{l} 0 < 0 \\ \vdash \\ 0 < n \end{array}$$

ARITH

$$\begin{array}{l} \bot \\ \vdash \\ 0 < n \end{array}$$

CNTR

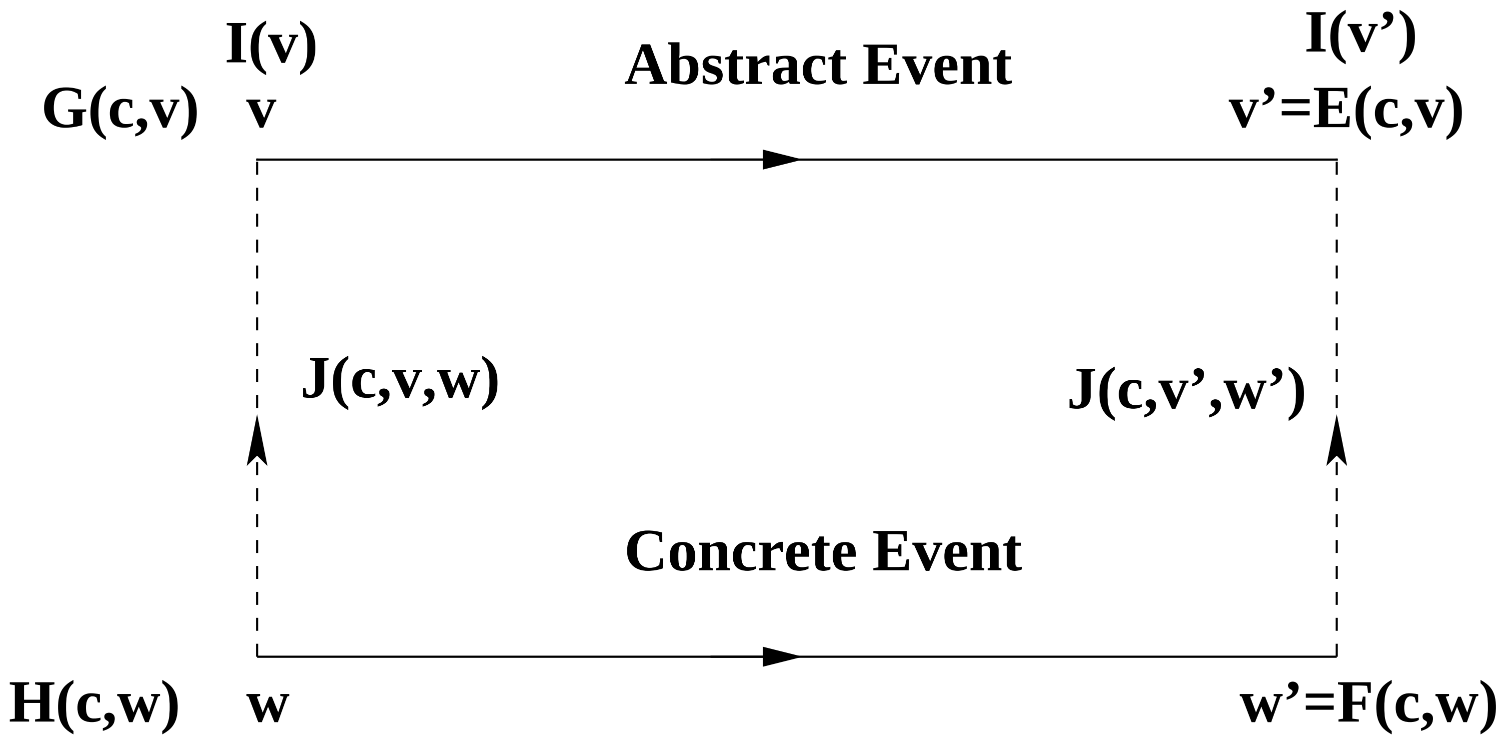
An Additional Rule: the Contradiction Rule

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

- It says that a false hypothesis entails any goal

$$\frac{}{\bot \ \vdash \ \mathbf{P}} \quad \text{CNTR}$$

Correctness of Invariant Refinement



Proof Obligation: Invariant Refinement

Axioms Abstract Invariants Concrete Invariants Concrete Guards $\vdash$ Modified Concrete Invariant	$\boldsymbol{A(c)}$ $\boldsymbol{I(c,v)}$ $\boldsymbol{J(c,v,w)}$ $\boldsymbol{H(c,w)}$ $\vdash$ $\boldsymbol{J_j(c,E(c,v),F(c,w))}$	INV

Overview of Proof Obligations

- 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

Applying Invariant Refinement 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

⊢

Modified Invariant **inv1_4**

$$
\begin{array}{l}
d \in \mathbb{N} \\
0 < d \\
n \in \mathbb{N} \\
n \leq d \\
a \in \mathbb{N} \\
b \in \mathbb{N} \\
c \in \mathbb{N} \\
a + b + c = n \\
a = 0 \quad \lor \quad c = 0 \\
a + b < d \\
c = 0 \\
\vdash \\
a + 1 + b + c = n + 1
\end{array}
$$

ML_out / **inv1_4** / INV

(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 ML_out / inv1_4 / INV

$$\begin{array}{l} d \in \mathbb{N} \\ 0 < d \\ n \in \mathbb{N} \\ n \leq d \\ a \in \mathbb{N} \\ b \in \mathbb{N} \\ c \in \mathbb{N} \\ a + b + c = n \\ a = 0 \quad \vee \quad c = 0 \\ a + b < d \\ c = 0 \\ \vdash \\ a + 1 + b + c = n + 1 \end{array}$$

MON

$$\begin{array}{l} a + b + c = n \\ \vdash \\ a + 1 + b + c = n + 1 \end{array}$$

ARITH ...

...

$$\begin{array}{l} a + b + c = n \\ \vdash \\ a + b + c + 1 = n + 1 \end{array}$$

EQ_LR

$$\vdash \quad n + 1 = n + 1$$

EQL

Applying Invariant Refinement 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

$\vdash$
Modified Invariant **inv1_5**

$$
\begin{array}{l}
d \in \mathbb{N} \\
0 < d \\
n \in \mathbb{N} \\
n \leq d \\
a \in \mathbb{N} \\
b \in \mathbb{N} \\
c \in \mathbb{N} \\
a + b + c = n \\
a = 0 \quad \lor \quad c = 0 \\
a + b < d \\
c = 0 \\
\vdash \\
a + 1 = 0 \ \lor \ c = 0
\end{array}
$$

ML_out / **inv1_5** / INV

(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 ML_out / inv1_5 / INV

$$
\begin{array}{l}
d \in \mathbb{N} \\
0 < d \\
n \in \mathbb{N} \\
n \leq d \\
a \in \mathbb{N} \\
b \in \mathbb{N} \\
c \in \mathbb{N} \\
a + b + c = n \\
a = 0 \lor c = 0 \\
a + b < d \\
c = 0 \\
\vdash \\
a + 1 = 0 \lor c = 0
\end{array}
$$

MON

$$
\begin{array}{l}
c = 0 \\
\vdash \\
a + 1 = 0 \lor c = 0
\end{array}
$$

OR_R2

$$
\begin{array}{l}
c = 0 \\
\vdash \\
c = 0
\end{array}
$$

HYP

Applying Invariant Refinement to Event ML_in

axm0_1
axm0_2
inv0_1
inv0_2
inv1_1
inv1_2
inv1_3
inv1_4
inv1_5
Concrete guards of ML_in
$\vdash$
Modified Invariant **inv1_4**

$$
\begin{array}{l}
d \in \mathbb{N} \\
0 < d \\
n \in \mathbb{N} \\
n \leq d \\
a \in \mathbb{N} \\
b \in \mathbb{N} \\
c \in \mathbb{N} \\
a + b + c = n \\
a = 0 \quad \lor \quad c = 0 \\
0 < c \\
\vdash \\
a + b + c - 1 = n - 1
\end{array}
$$

ML_in / **inv1_4** / INV

(abstract-)ML_in
when
$0 < n$
then
$n := n - 1$
end

(concrte-)ML_in
when
$0 < c$
then
$c := c - 1$
end

Proof of ML_in / inv1_4 / INV

$$
\begin{array}{l}
d \in \mathbb{N} \\
0 < d \\
n \in \mathbb{N} \\
n \leq d \\
a \in \mathbb{N} \\
b \in \mathbb{N} \\
c \in \mathbb{N} \\
a + b + c = n \\
a = 0 \;\lor\; c = 0 \\
0 < c \\
\vdash \\
a + b + c - 1 = n - 1
\end{array}
$$

MON

$$
\begin{array}{l}
a + b + c = n \\
\vdash \\
a + b + c - 1 = n - 1
\end{array}
$$

EQ_LR

$$\vdash \quad n - 1 = n - 1$$

EQL

Applying Invariant Refinement to Event ML_in

axm0_1
axm0_2
inv0_1
inv0_2
inv1_1
inv1_2
inv1_3
inv1_4
inv1_5
Concrete guards of ML_in
$\vdash$
Modified Invariant **inv1_5**

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

ML_in / **inv1_5** / INV

(abstract-)ML_in
when
$0 < n$
then
$n := n - 1$
end

(concrete-)ML_in
when
$0 < c$
then
$c := c - 1$
end

Proof of ML_in / inv1_5 / INV

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

MON

$$\begin{array}{l} a = 0 \ \lor \ c = 0 \\ 0 < c \\ \vdash \\ a = 0 \ \lor \ c - 1 = 0 \end{array}$$

OR_L

$$\begin{array}{l} a = 0 \\ 0 < c \\ \vdash \\ a = 0 \ \lor \ c - 1 = 0 \end{array}$$

MON $\cdots$

$$\begin{array}{l} c = 0 \\ 0 < c \\ \vdash \\ a = 0 \ \lor \ c - 1 = 0 \end{array}$$

EQ_LR $\cdots$

Proof of ML_in / inv1_5 / INV

$$
\begin{array}{l}
d \in \mathbb{N} \\
0 < d \\
n \in \mathbb{N} \\
n \leq d \\
a \in \mathbb{N} \\
b \in \mathbb{N} \\
c \in \mathbb{N} \\
a + b + c = n \\
a = 0 \;\lor\; c = 0 \\
0 < c \\
\vdash \\
a = 0 \;\lor\; c - 1 = 0
\end{array}
$$

MON

$$
\begin{array}{l}
a = 0 \;\lor\; c = 0 \\
0 < c \\
\vdash \\
a = 0 \;\lor\; c - 1 = 0
\end{array}
$$

OR_L

$$
\begin{array}{l}
a = 0 \\
0 < c \\
\vdash \\
a = 0 \;\lor\; c - 1 = 0
\end{array}
$$

MON $\cdots$

$$
\begin{array}{l}
c = 0 \\
0 < c \\
\vdash \\
a = 0 \;\lor\; c - 1 = 0
\end{array}
$$

EQ_LR $\cdots$

$\cdots$

$$
\begin{array}{l}
a = 0 \\
\vdash \\
a = 0 \;\lor\; c - 1 = 0
\end{array}
$$

OR_R1

$$a = 0 \;\vdash\; a = 0$$

HYP

$\cdots$

$$0 < 0 \;\vdash\; a = 0 \;\lor\; -1 = 0$$

ARITH

$$\bot \;\vdash\; a = 0 \;\lor\; -1 = 0$$

CNTR

Refining the Initialization Event init

- Concrete initialization

init
$$a := 0$$
$$b := 0$$
$$c := 0$$

- Corresponding after predicate

$$a' = 0 \quad \wedge \quad b' = 0 \quad \wedge \quad c' = 0$$

Proof Obligation: Initialization Refinement

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 $w' = L(c)$

Axioms $\vdash$ Modified concrete invariants	$A(c)$ $\vdash$ $J_j(c, K(c), L(c))$	INV

Overview of Proof Obligations

- 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 PO

axm0_1
axm0_2
$\vdash$
Modified concrete invariant **inv1_4**

$$(a + b + c = n)$$

$$d \in \mathbb{N}$$
$$d > 0$$
$\vdash$
$$0 + 0 + 0 = 0$$

axm0_1
axm0_2
$\vdash$
Modified concrete invariant **inv1_5**

$$(a = 0 \;\vee\; c = 0)$$

$$d \in \mathbb{N}$$
$$d > 0$$
$\vdash$
$$0 = 0 \;\vee\; 0 = 0$$

Adding New Events

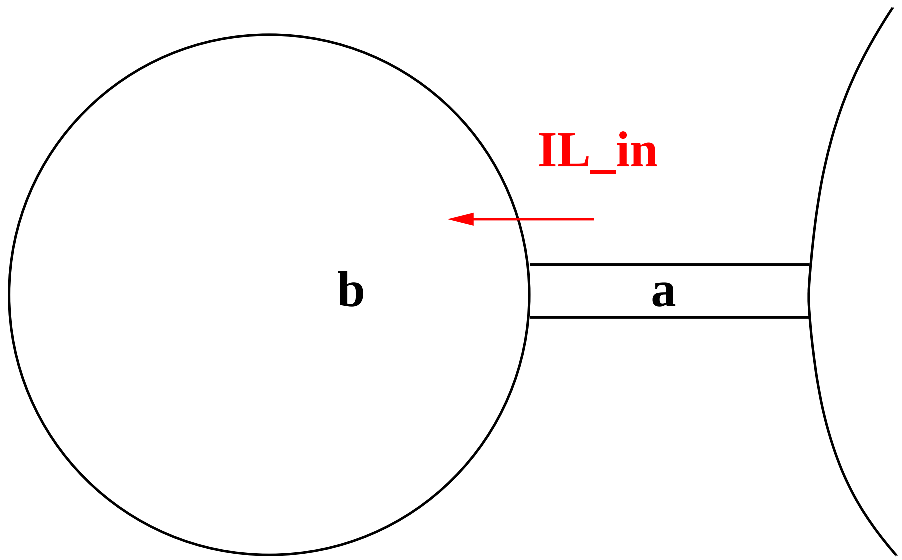
- new events add transitions that have no abstract counterpart

- can be seen as a kind of internal steps (w.r.t. abstract model)

- can only be seen by an observer who is “zooming in”

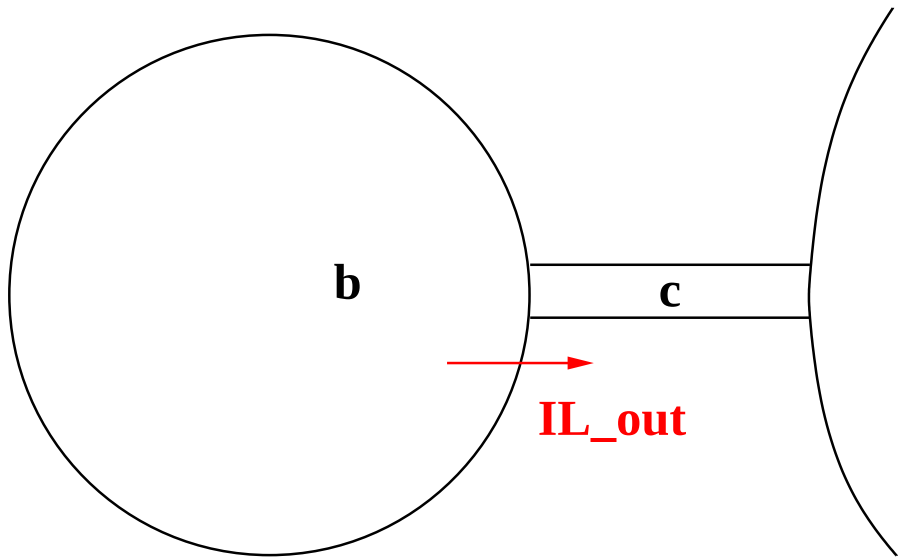
- temporal refinement: refined model has a finer time granularity

New Event IL_in



IL_in
when
$0 < a$
then
$a := a - 1$
$b := b + 1$
end

New Event IL_out



IL_out
  when
    $0 < b$
    $a = 0$
  then
    $b := b - 1$
    $c := c + 1$
  end

Several Actions Done Together

IL_in
  when
    $0 < a$
  then
    $a := a - 1$
    $b := b + 1$
  end

IL_out
  when
    $0 < b$
    $a = 0$
  then
    $b := b - 1$
    $c := c + 1$
  end

Before-after predicates

$$a' = a + 1 \quad \wedge \quad b' = b + 1 \quad \wedge \quad c' = c$$

$$a' = a \quad \wedge \quad b' = b - 1 \quad \wedge \quad c' = c + 1$$

The empty assignment: skip

The before-after predicate of skip in the initial model

$$n' = n$$

The before-after predicate of skip in the first refinement

$$a' = a \quad \wedge \quad b' = b \quad \wedge \quad c' = c$$

The guard of the skip event is true.

Refinement Proof Obligations for New Events

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

- Guard strengthening is trivial
- Need to prove invariant refinement

(2) The new events must not diverge

- To prove this we have to exhibit a variant
- The variant yields a natural number (could be more complex)
- Each new event must decrease this variant

Overview of Proof Obligations

- 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 IL_in Refines skip (1)

axm0_1
axm0_2
inv0_1
inv0_2
inv1_1
inv1_2
inv1_3
inv1_4
inv1_5
Concrete guards of IL_in
$\vdash$
Modified Invariant **inv1_4**

$$
\begin{array}{l}
d \in \mathbb{N} \\
0 < d \\
n \in \mathbb{N} \\
n \leq d \\
a \in \mathbb{N} \\
b \in \mathbb{N} \\
c \in \mathbb{N} \\
a + b + c = n \\
a = 0 \quad \lor \quad c = 0 \\
0 < a \\
\vdash \\
a - 1 + b + 1 + c = n
\end{array}
$$

IL_in / **inv1_4** / INV

IL_in
when
$0 < a$
then
$a := a - 1$
$b := b + 1$
end

Proof of IL_in / inv1_4 / INV

$$
\begin{array}{l}
\quad d \in \mathbb{N} \\
\quad 0 < d \\
\quad n \in \mathbb{N} \\
\quad n \leq d \\
\quad a \in \mathbb{N} \\
\quad b \in \mathbb{N} \\
\quad c \in \mathbb{N} \\
\quad a + b + c = n \\
\quad a = 0 \;\vee\; c = 0 \\
\quad 0 < a \\
\vdash \\
\quad a - 1 + b + 1 + c = n
\end{array}
$$

MON

$$
\begin{array}{l}
\quad a + b + c = n \\
\vdash \\
\quad a - 1 + b + 1 + c = n
\end{array}
$$

ARITH

$$
\begin{array}{l}
\quad a + b + c = n \\
\vdash \\
\quad a + b + c = n
\end{array}
$$

HYP

Event IL_in Refines skip (2)

axm0_1
axm0_2
inv0_1
inv0_2
inv1_1
inv1_2
inv1_3
inv1_4
inv1_5
Concrete guards of IL_in
⊢
Modified Invariant **inv1_5**

$$
\begin{array}{l}
d \in \mathbb{N} \\
0 < d \\
n \in \mathbb{N} \\
n \leq d \\
a \in \mathbb{N} \\
b \in \mathbb{N} \\
c \in \mathbb{N} \\
a + b + c = n \\
a = 0 \;\vee\; c = 0 \\
0 < a \\
\vdash \\
a - 1 = 0 \;\vee\; c = 0
\end{array}
$$

IL_in / **inv1_5** / INV

IL_in
  when
    $0 < a$
  then
    $a := a - 1$
    $b := b + 1$
  end

Proof of IL_in / inv1_5 / INV

$$
\begin{array}{l}
d \in \mathbb{N} \\
0 < d \\
n \in \mathbb{N} \\
n \leq d \\
a \in \mathbb{N} \\
b \in \mathbb{N} \\
c \in \mathbb{N} \\
a + b + c = n \\
a = 0 \;\vee\; c = 0 \\
0 < a \\
\vdash \\
a - 1 = 0 \;\vee\; c = 0
\end{array}
\quad \text{MON} \quad
\begin{array}{l}
a = 0 \;\vee\; c = 0 \\
0 < a \\
\vdash \\
a - 1 = 0 \;\vee\; c = 0
\end{array}
\quad \text{OR\_L} \quad \cdots
$$

Proof of IL_in / inv1_5 / INV

$$\begin{array}{l} d \in \mathbb{N} \\ 0 < d \\ n \in \mathbb{N} \\ n \leq d \\ a \in \mathbb{N} \\ b \in \mathbb{N} \\ c \in \mathbb{N} \\ a + b + c = n \\ a = 0 \;\vee\; c = 0 \\ 0 < a \\ \vdash \\ a - 1 = 0 \;\vee\; c = 0 \end{array}$$

MON

$$\begin{array}{l} a = 0 \;\vee\; c = 0 \\ 0 < a \\ \vdash \\ a - 1 = 0 \;\vee\; c = 0 \end{array}$$

OR_L $\cdots$

$\cdots$

$$\begin{array}{l} a = 0 \\ 0 < a \\ \vdash \\ a - 1 = 0 \;\vee\; c = 0 \end{array}$$

EQ_LR

$$\begin{array}{l} 0 < 0 \\ \vdash \\ -1 = 0 \;\vee\; c = 0 \end{array}$$

ARITH

$$\begin{array}{l} \bot \\ \vdash \\ -1 = 0 \;\vee\; c = 0 \end{array}$$

CNTR

$$\begin{array}{l} c = 0 \\ 0 < a \\ \vdash \\ a - 1 = 0 \;\vee\; c = 0 \end{array}$$

MON

$$\begin{array}{l} c = 0 \\ \vdash \\ a - 1 = 0 \;\vee\; c = 0 \end{array}$$

OR_R2

$$c = 0 \;\vdash\; c = 0$$

HYP

Proof Obligation: Convergence of New Events (1)

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 Abstract invariants Concrete invariants Concrete guard of a new event $\vdash$ Variant $\in \mathbb{N}$	$A(c)$ $I(c,v)$ $J(c,v,w)$ $H(c,w)$ $\vdash$ $V(c,w) \in \mathbb{N}$	NAT

Proof Obligation: Convergence of New Events (2)

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 $V(c,w)$

Axioms Abstract invariants Concrete invariants Concrete guard $\vdash$ Modified Var. $<$ Var.	$A(c)$ $I(c,v)$ $J(c,v,w)$ $H(c,w)$ $\vdash$ $V(c,F(c,w)) < V(c,w)$	VAR

Proposed Variant

variant_1: $2 * a + b$

- Weighted sum of a and b

Overview of Proof Obligations

- 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

- IL_in / NAT
- IL_out / NAT
- IL_in / VAR
- IL_out / VAR

Decreasing of the Variant by Event IL_in

axm0_1
axm0_2
inv0_1
inv0_2
inv1_1
inv1_2
inv1_3
inv1_4
inv1_5
Concrete guard of IL_in
⊢
Modified variant < Variant

$$
\begin{array}{l}
d \in \mathbb{N} \\
0 < d \\
n \in \mathbb{N} \\
n \leq d \\
a \in \mathbb{N} \\
b \in \mathbb{N} \\
c \in \mathbb{N} \\
a + b + c = n \\
a = 0 \quad \lor \quad c = 0 \\
0 < a \\
\vdash \\
2 * (a - 1) + b + 1 < 2 * a + b
\end{array}
$$

IL_in / VAR

IL_in
when
$0 < a$
then
$a := a - 1$
$b := b + 1$
end

Decreasing of the Variant by Event IL_out

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

⊢
Modified variant < Variant

$$
\begin{array}{l}
d \in \mathbb{N} \\
0 < d \\
n \in \mathbb{N} \\
n \leq d \\
a \in \mathbb{N} \\
b \in \mathbb{N} \\
c \in \mathbb{N} \\
a + b + c = n \\
a = 0 \quad \lor \quad c = 0 \\
0 < b \\
a = 0 \\
\vdash \\
2 * a + b - 1 < 2 * a + b
\end{array}
$$

IL_out / VAR

IL_out
 when
  $0 < b$
  $a = 0$
 then
  $b := b - 1$
  $c := c + 1$
 end

Relative Deadlock Freedom

There a no new deadlocks in the concrete model, that is, all deadlocks of the concrete model are already present in the abstract model.

Proof obligation requires that whenever some abstract event is enabled then so is some concrete event.

This proof obligaiton is optional (depending on system under study).

Proof Obligation: Relative Deadlock Freedom

The $G_i(c, v)$ are the abstract guards

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

If some abstract guard is true then so is some concrete guard:

$A(c)$ $I(c, v)$ $J(c, v, w)$ $G_1(c, v) \ \vee \ \ldots \ \vee \ G_m(c, v)$ $\vdash$ $H_1(c, w) \ \vee \ \ldots \ \vee \ H_n(c, w)$	DLF

Applying the Relative Deadlock Freedom PO

axm0_1
axm0_2
inv0_1
inv0_2
inv1_1
inv1_2
inv1_3
inv1_4
inv1_5
Disjunction of abstract guards
$\vdash$
Disjunction of concrete guards

$$
\begin{array}{l}
d \in \mathbb{N} \\
0 < d \\
n \in \mathbb{N} \\
n \leq d \\
a \in \mathbb{N} \\
b \in \mathbb{N} \\
c \in \mathbb{N} \\
a + b + c = n \\
a = 0 \;\lor\; c = 0 \\
0 < n \;\lor\; n < d \\
\vdash \\
(a + b < d \;\land\; c = 0) \;\lor \\
c > 0 \;\lor\; a > 0 \\
(b > 0 \;\land\; a = 0)
\end{array}
$$

DLF

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

ML_in
when
$c > 0$
then
$c := c - 1$
end

IL_in
when
$a > 0$
then
$a := a - 1$
$b := b + 1$
end

IL_out
when
$b > 0$
$a = 0$
then
$b := b - 1$
$c := c + 1$
end

More Inference Rules: Negation and Conjunction

$$\frac{\mathbf{H}, \neg\, \mathbf{P} \;\;\vdash\;\; \mathbf{Q}}{\mathbf{H} \;\;\vdash\;\; \mathbf{P} \;\vee\; \mathbf{Q}} \quad \text{NEG}$$

$$\frac{\mathbf{H}, \mathbf{P}, \mathbf{Q} \;\;\vdash\;\; \mathbf{R}}{\mathbf{H},\; \mathbf{P} \wedge \mathbf{Q} \;\;\vdash\;\; \mathbf{R}} \quad \text{AND\_L}$$

$$\frac{\mathbf{H} \;\;\vdash\;\; \mathbf{P} \qquad \mathbf{H} \;\;\vdash\;\; \mathbf{Q}}{\mathbf{H} \;\;\vdash\;\; \mathbf{P} \wedge \mathbf{Q}} \quad \text{AND\_R}$$

Proof of DLF

$d \in \mathbb{N}$
$0 < d$
$n \in \mathbb{N}$
$n \leq d$
$a \in \mathbb{N}$
$b \in \mathbb{N}$
$c \in \mathbb{N}$
$a + b + c = n$
$a = 0 \;\vee\; c = 0$
$n > 0 \;\vee\; n < d$
$\vdash$
$(a + b < d \;\wedge\; c = 0) \;\vee$
$c > 0 \;\vee$
$a > 0 \;\vee$
$(b > 0 \;\wedge\; a = 0)$

MON

$a \in \mathbb{N}$
$c \in \mathbb{N}$
$a + b + c = n$
$n > 0 \;\vee\; n < d$
$\vdash$
$(a + b < d \;\wedge\; c = 0) \;\vee$
$c > 0 \;\vee$
$a > 0 \;\vee$
$(b > 0 \;\wedge\; a = 0)$

NEG

$a \in \mathbb{N}$
$c \in \mathbb{N}$
$a + b + c = n$
$n > 0 \;\vee\; n < d$
$\neg\,(c > 0)$
$\vdash$
$(a + b < d \;\wedge\; c = 0) \;\vee$
$a > 0 \;\vee$
$(b > 0 \;\wedge\; a = 0)$

ARITH $\cdots$

Proof of DLF

$$
\begin{array}{l}
d \in \mathbb{N} \\
0 < d \\
n \in \mathbb{N} \\
n \leq d \\
a \in \mathbb{N} \\
b \in \mathbb{N} \\
c \in \mathbb{N} \\
a + b + c = n \\
a = 0 \ \vee \ c = 0 \\
n > 0 \ \vee \ n < d \\
\vdash \\
(a + b < d \ \wedge \ c = 0) \ \vee \\
c > 0 \ \vee \\
a > 0 \ \vee \\
(b > 0 \ \wedge \ a = 0)
\end{array}
$$

MON

$$
\begin{array}{l}
a \in \mathbb{N} \\
c \in \mathbb{N} \\
a + b + c = n \\
n > 0 \ \vee \ n < d \\
\vdash \\
(a + b < d \ \wedge \ c = 0) \ \vee \\
c > 0 \ \vee \\
a > 0 \ \vee \\
(b > 0 \ \wedge \ a = 0)
\end{array}
$$

NEG

$$
\begin{array}{l}
a \in \mathbb{N} \\
c \in \mathbb{N} \\
a + b + c = n \\
n > 0 \ \vee \ n < d \\
\neg (c > 0) \\
\vdash \\
(a + b < d \ \wedge \ c = 0) \ \vee \\
a > 0 \ \vee \\
(b > 0 \ \wedge \ a = 0)
\end{array}
$$

ARITH $\cdots$

$\cdots$

$$
\begin{array}{l}
a \in \mathbb{N} \\
a + b + c = n \\
n > 0 \ \vee \ n < d \\
c = 0 \\
\vdash \\
(a + b < d \ \wedge \ c = 0) \ \vee \\
a > 0 \ \vee \\
(b > 0 \ \wedge \ a = 0)
\end{array}
$$

EQ_LR

$$
\begin{array}{l}
a \in \mathbb{N} \\
a + b + 0 = n \\
n > 0 \ \vee \ n < d \\
\vdash \\
(a + b < d \ \wedge \ 0 = 0) \ \vee \\
a > 0 \ \vee \\
(b > 0 \ \wedge \ a = 0)
\end{array}
$$

NEG

$$
\begin{array}{l}
a \in \mathbb{N} \\
a + b + 0 = n \\
n > 0 \ \vee \ n < d \\
\neg (a > 0) \\
\vdash \\
(a + b < d \ \wedge \ 0 = 0) \ \vee \\
(b > 0 \ \wedge \ a = 0)
\end{array}
$$

ARITH $\cdots$

Proof of DLF (cont'd)

$$\cdots \quad \begin{array}{l} a + b + 0 = n \\ n > 0 \ \vee \ n < d \\ a = 0 \\ \vdash \\ (a + b < d \ \wedge \ 0 = 0) \ \vee \\ (b > 0 \ \wedge \ a = 0) \end{array} \quad \text{EQ\_LR} \quad \begin{array}{l} 0 + b + 0 = n \\ n > 0 \ \vee \ n < d \\ \vdash \\ (0 + b < d \ \wedge \ 0 = 0) \ \vee \\ (b > 0 \ \wedge \ 0 = 0) \end{array} \quad \text{ARITH} \quad \begin{array}{l} b = n \\ n > 0 \ \vee \ n < d \\ \vdash \\ (b < d \ \wedge \ 0 = 0) \ \vee \\ (b > 0 \ \wedge \ 0 = 0) \end{array} \quad \text{EQ\_LR} \quad \cdots$$

Proof of DLF (cont'd)

$$\ldots \quad \begin{array}{l} a+b+0=n \\ n>0 \;\vee\; n<d \\ a=0 \\ \vdash \\ (a+b<d \;\wedge\; 0=0) \;\vee \\ (b>0 \;\wedge\; a=0) \end{array} \quad \text{EQ\_LR} \quad \begin{array}{l} 0+b+0=n \\ n>0 \;\vee\; n<d \\ \vdash \\ (0+b<d \;\wedge\; 0=0) \;\vee \\ (b>0 \;\wedge\; 0=0) \end{array} \quad \text{ARITH} \quad \begin{array}{l} b=n \\ n>0 \;\vee\; n<d \\ \vdash \\ (b<d \;\wedge\; 0=0) \;\vee \\ (b>0 \;\wedge\; 0=0) \end{array} \quad \text{EQ\_LR} \quad \ldots$$

$$\ldots \quad \begin{array}{l} n>0 \;\vee\; n<d \\ \vdash \\ (n<d \;\wedge\; 0=0) \;\vee \\ (n>0 \;\wedge\; 0=0) \end{array} \quad \text{OR\_L}$$

$$\begin{array}{l} n>0 \\ \vdash \\ (n<d \;\wedge\; 0=0) \;\vee \\ (n>0 \;\wedge\; 0=0) \end{array} \quad \text{OR\_R2} \quad \begin{array}{l} n>0 \\ \vdash \\ n>0 \;\wedge\; 0=0 \end{array} \quad \text{AND\_R} \quad \left\{ \begin{array}{l} n>0 \vdash n>0 \quad \text{HYP} \\ n>0 \vdash 0=0 \quad \text{EQL} \end{array} \right.$$

$$\begin{array}{l} n<d \\ \vdash \\ (n<d \;\wedge\; 0=0) \;\vee \\ (n>0 \;\wedge\; 0=0) \end{array} \quad \text{OR\_R1} \quad \begin{array}{l} n<d \\ \vdash \\ n<d \;\wedge\; 0=0 \end{array} \quad \text{AND\_R} \quad \left\{ \begin{array}{l} n<d \vdash n<d \quad \text{HYP} \\ n<d \vdash 0=0 \quad \text{EQL} \end{array} \right.$$

Overview of Proof Obligations

- 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
- IL_in / NAT done
- IL_out / NAT done
- IL_in / VAR done
- IL_out / VAR done
- DLF done

Summary of Refinement POs

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

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

- For all events:
    - Relative deadlock freedom: DLF

Proof Obligations for Refinement (1/2)

Axioms Abstract invariants Concrete invariants Concrete guards $\vdash$ Abstract guard	GRD

Axioms Abstract invariants Concrete invariants Concrete guard $\vdash$ Modified concrete invariant	INV

Axioms $\vdash$ Modified concrete invariant	INV

Proof Obligations for Refinement (2/2)

Axioms Abstract invariants Concrete invariants Concrete guards of a new event $\vdash$ Variant $\in \mathbb{N}$	NAT

Axioms Abstract invariants Concrete invariants Concrete guards of a new event $\vdash$ Modified variant $<$ Variant	VAR

Axioms Abstract invariants Concrete invariants Disjunction of abstract events guards $\vdash$ Disjunction of concrete events guards	DLF

State of the First Refinement

constants: d

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

variant1: $2 * a + b$

Events of the First Refinement

init
$a := 0$
$b := 0$
$c := 0$

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

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

IL_in
when
$0 < a$
then
$a := a - 1$
$b := b + 1$
end

IL_out
when
$0 < b$
$a = 0$
then
$b := b - 1$
$c := c + 1$
end

Our Refinement Strategy

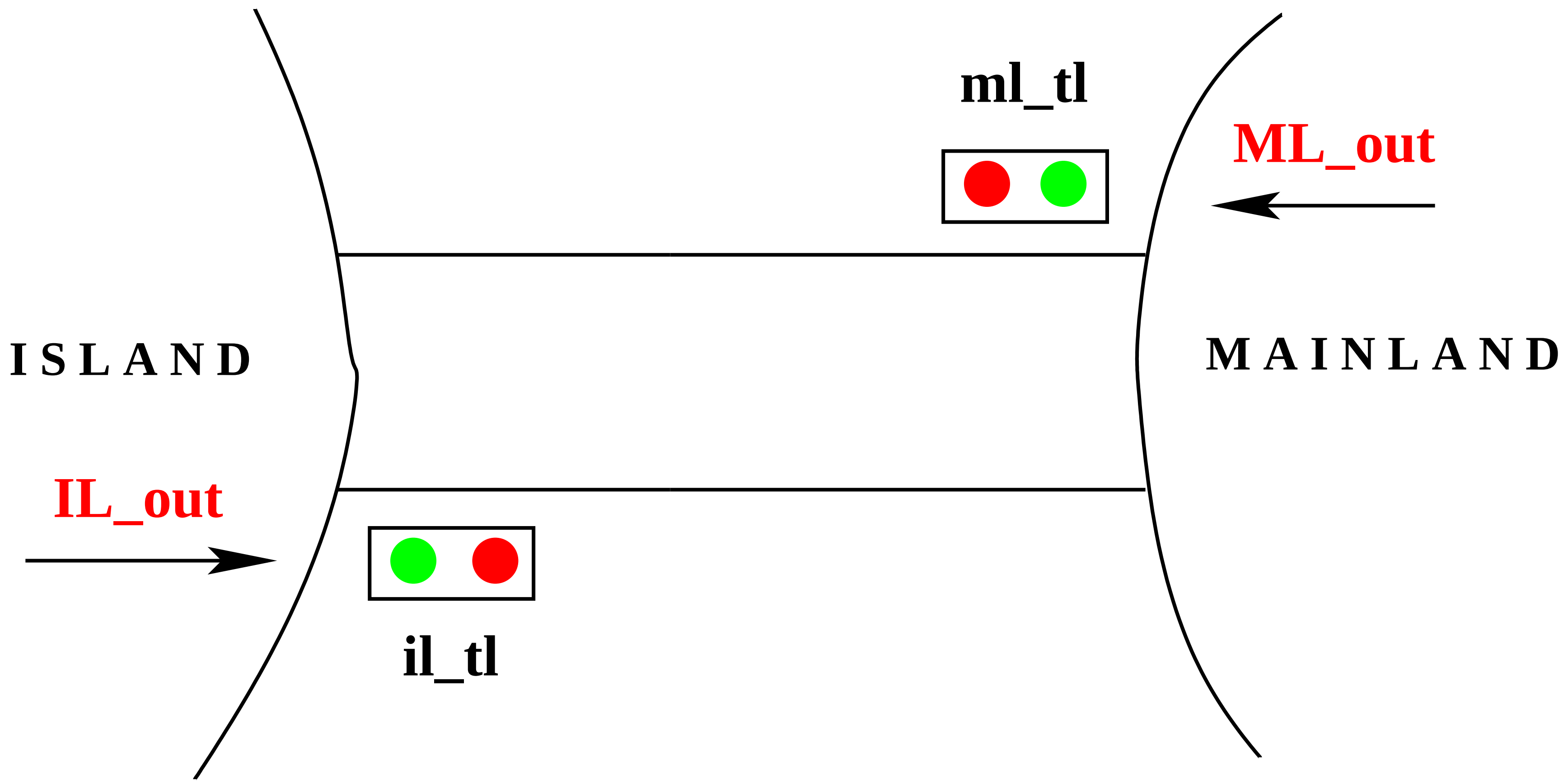
- Initial model: Limiting the number of cars (FUN-2)

- 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: $red, green$

axm2_1: $COLOR = \{green, red\}$

axm2_2: $green \neq red$

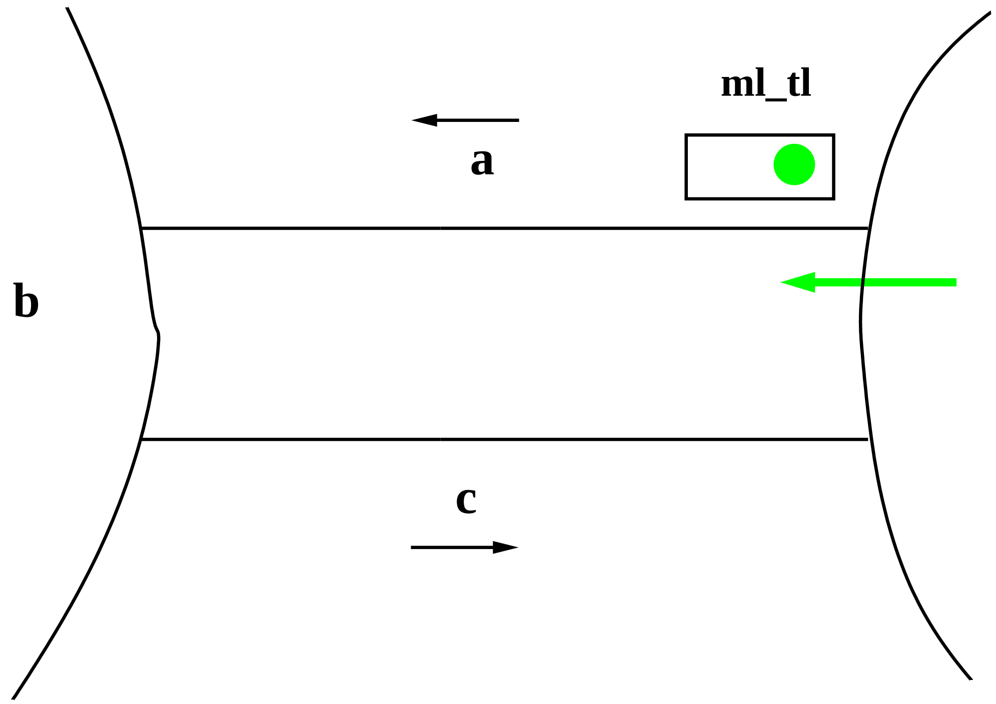
Extending the Variables

$$il\_tl \in COLOR$$

$$ml\_tl \in COLOR$$

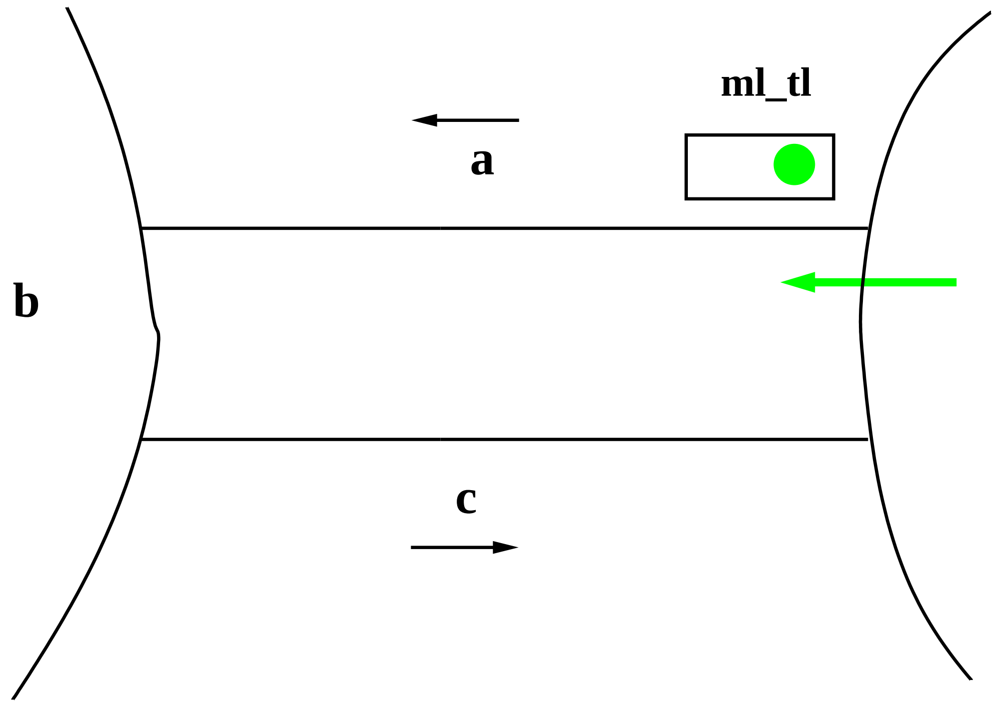
Remark: Events IL_in and ML_in are not modified in this refinement

Extending the Invariant (1)



- A green mainland traffic light implies safe access to the bridge

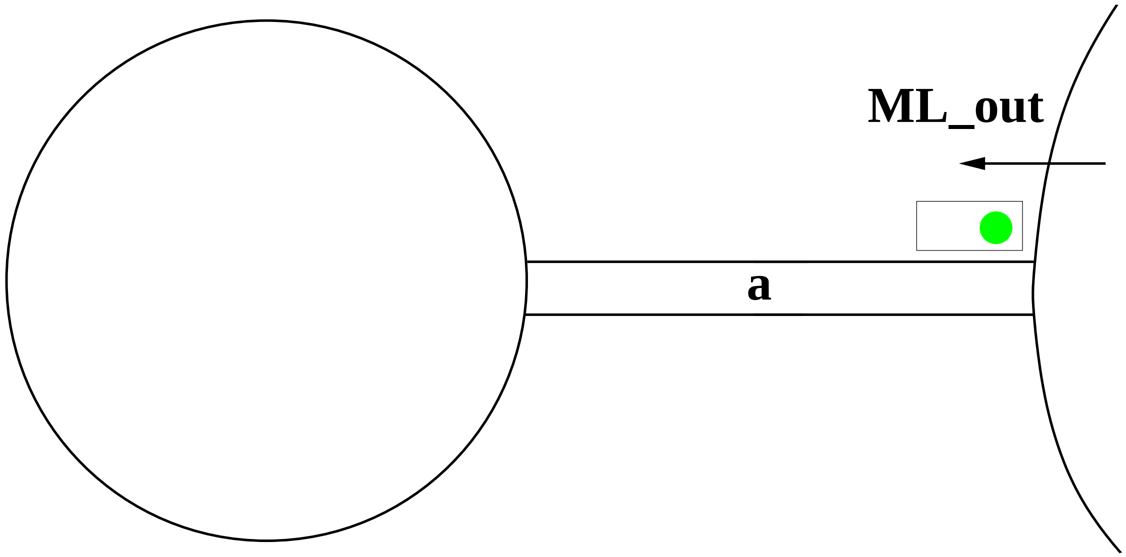
Extending the Invariant (1)



- A green mainland traffic light implies safe access to the bridge

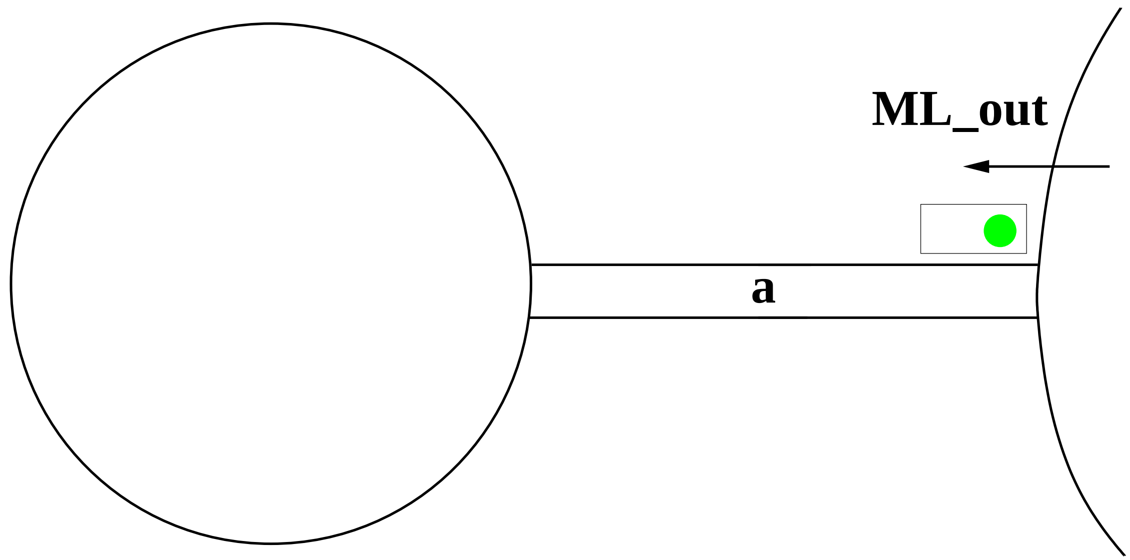
$$ml\_tl = \textbf{green} \Rightarrow c = 0 \wedge a + b < d$$

Refining Event ML_out



(abstract_)ML_out
when
$c = 0$
$a + b < d$
then
$a := a + 1$
end

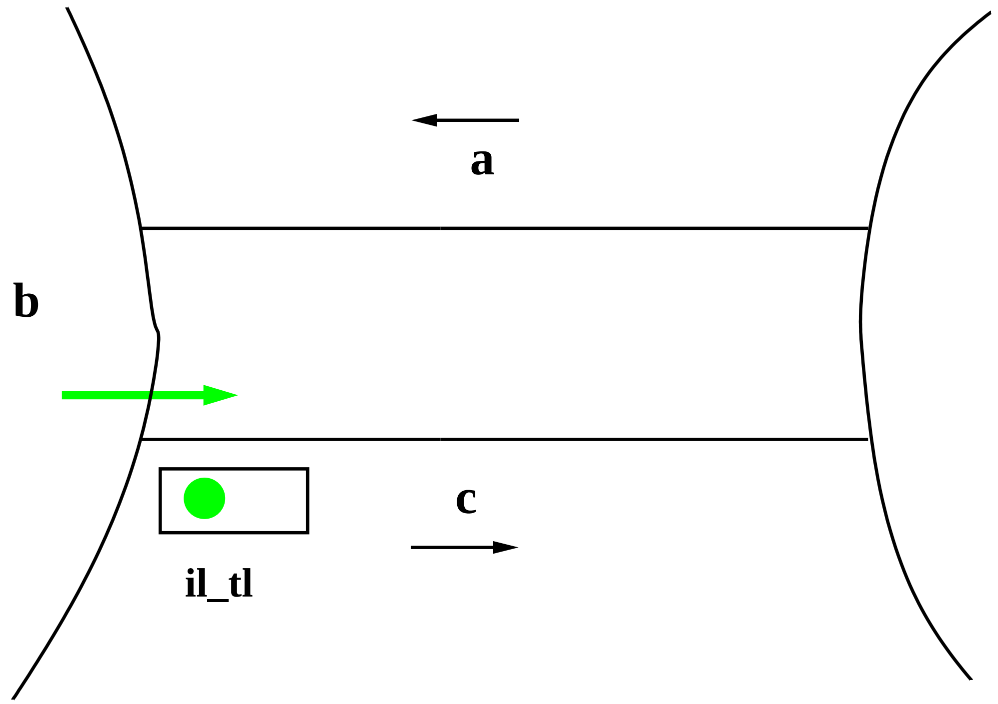
Refining Event ML_out



(abstract_)ML_out
when
$c = 0$
$a + b < d$
then
$a := a + 1$
end

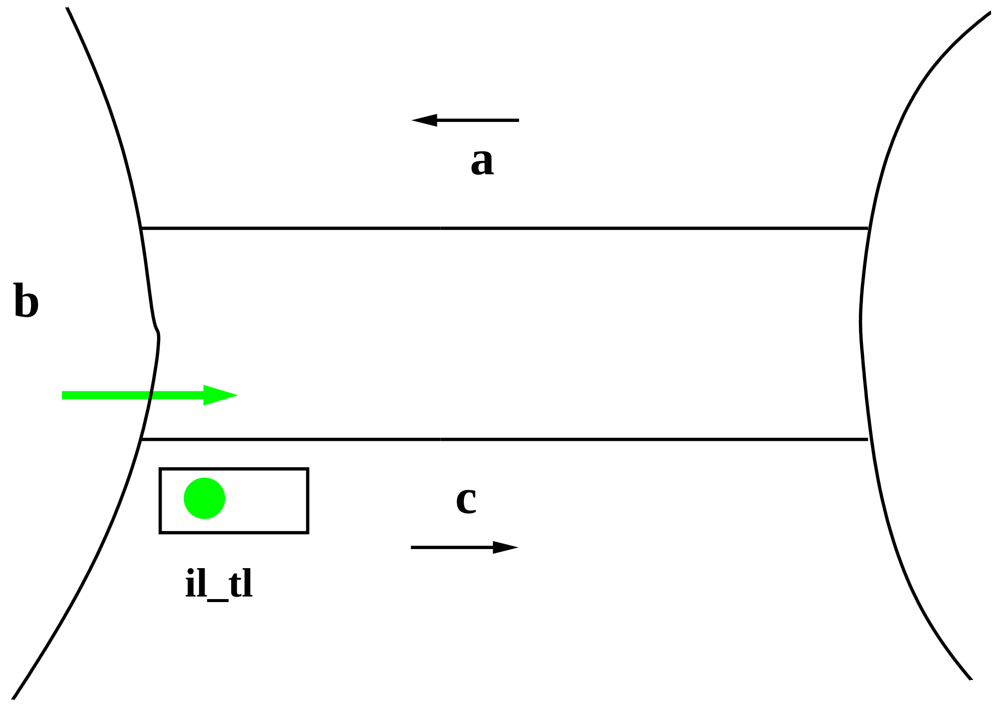
(concrete_)ML_out
when
$ml\_tl = \textbf{green}$
then
$a := a + 1$
end

Extending the Invariant (2)



- A green island traffic light implies safe access to the bridge

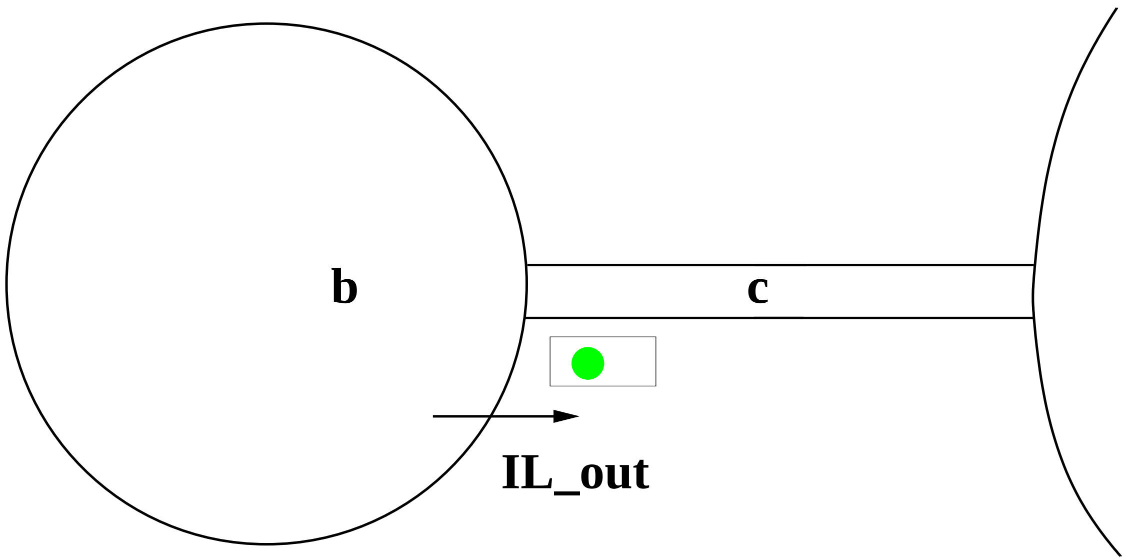
Extending the Invariant (2)



- A green island traffic light implies safe access to the bridge

$$il\_tl = \mathbf{green} \;\Rightarrow\; a = 0 \;\wedge\; 0 < b$$

Refining Event IL_out



(abstract_)IL_out

when

$a = 0$

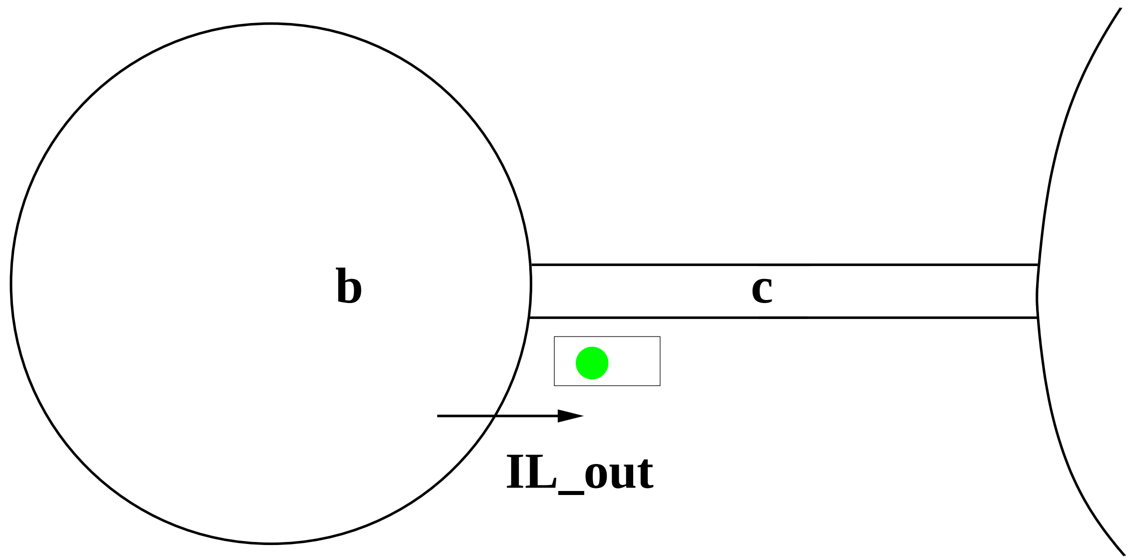
$0 < b$

then

$b, c := b - 1, c + 1$

end

Refining Event IL_out



(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

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

IL_tl_green
when
$il\_tl = \textbf{red}$
$a = 0$
$0 < b$
then
$il\_tl := \textbf{green}$
end

- Turning lights to green when proper conditions hold

Summary of State Refinement (so far)

variables: a, b, c, ml_tl, il_tl

inv2_1: $ml\_tl \in COLOR$

inv2_2: $il\_tl \in COLOR$

inv2_3: $ml\_tl = \text{green} \Rightarrow a + b < d \wedge c = 0$

inv2_4: $il\_tl = \text{green} \Rightarrow 0 < b \wedge a = 0$

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
```

Superposition

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

Superposition: Introduction of a new Rule

Abstract_Event	Concrete_Event
Abstract_Event **when** $G(c, u, v)$ **then** $u := E(c, u, v)$ $v := M(c, u, v)$ **end**	Concrete_Event **when** $H(c, v, w)$ **then** $v := N(c, v, w)$ $w := F(c, v, w)$ **end**

Axioms Abstract invariants Concrete invariants Concrete guards $\Rightarrow$ Same actions on common variables	$A(c)$ $I(c, u, v)$ $J(c, u, v, w)$ $H(c, v, w)$ $\Rightarrow$ $M(c, u, v) = N(c, v, w)$	SIM

Proving the Refinement of the Four Old Events

- 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:

inv2_3: $ml\_tl = \textbf{green} \;\Rightarrow\; a + b < d \;\wedge\; c = 0$

inv2_4: $il\_tl = \textbf{green} \;\Rightarrow\; 0 < b \;\wedge\; a = 0$

Proving the Refinement of the Four Old Events

(abstract-)ML_out
when
$c = 0$
$a + b < d$
then
$a := a + 1$
end

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

(abstract-)ML_in
when
$0 < c$
then
$c := c - 1$
end

(abstract-)IL_in
when
$0 < a$
then
$a := a - 1$
$b := b + 1$
end

(concrete-)ML_out
when
$ml\_tl = \textbf{green}$
then
$a := a + 1$
end

(concrete-)IL_out
when
$il\_tl = \textbf{green}$
then
$b := b - 1$
$c := c + 1$
end

(concrete-)ML_in
when
$0 < c$
then
$c := c - 1$
end

(concrete-)IL_in
when
$0 < a$
then
$a := a - 1$
$b := b + 1$
end

- SIM is completely trivial since the actions are the same

Proving the Refinement of the Four Old Events

(abstract-)ML_out
when
$c = 0$
$a + b < d$
then
$a := a + 1$
end

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

(abstract-)ML_in
when
$0 < c$
then
$c := c - 1$
end

(abstract-)IL_in
when
$0 < a$
then
$a := a - 1$
$b := b + 1$
end

(concrete-)ML_out
when
$ml\_tl = \textbf{green}$
then
$a := a + 1$
end

(concrete-)IL_out
when
$il\_tl = \textbf{green}$
then
$b := b - 1$
$c := c + 1$
end

(concrete-)ML_in
when
$0 < c$
then
$c := c - 1$
end

(concrete-)IL_in
when
$0 < a$
then
$a := a - 1$
$b := b + 1$
end

- GRD is also trivial

inv2_3: $ml\_tl = \textbf{green} \Rightarrow a + b < d \wedge c = 0$

inv2_4: $il\_tl = \textbf{green} \Rightarrow 0 < b \wedge a = 0$

Proving the Refinement of the Four Old Events

(abstract-)ML_out
when
$c = 0$
$a + b < d$
then
$a := a + 1$
end

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

(abstract-)ML_in
when
$0 < c$
then
$c := c - 1$
end

(abstract-)IL_in
when
$0 < a$
then
$a := a - 1$
$b := b + 1$
end

(concrete-)ML_out
when
$ml\_tl = \mathbf{green}$
then
$a := a + 1$
end

(concrete-)IL_out
when
$il\_tl = \mathbf{green}$
then
$b := b - 1$
$c := c + 1$
end

(concrete-)ML_in
when
$0 < c$
then
$c := c - 1$
end

(concrete-)IL_in
when
$0 < a$
then
$a := a - 1$
$b := b + 1$
end

- INV applied to ML_in and IL_in holds trivially

inv2_3: $ml\_tl = \mathbf{green} \Rightarrow a + b < d \land c = 0$

inv2_4: $il\_tl = \mathbf{green} \Rightarrow 0 < b \land a = 0$

Proving the Refinement of the Four Old Events

(abstract-)ML_out
when
$c = 0$
$a + b < d$
then
$a := a + 1$
end

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

(abstract-)ML_in
when
$0 < c$
then
$c := c - 1$
end

(abstract-)IL_in
when
$0 < a$
then
$a := a - 1$
$b := b + 1$
end

(concrete-)ML_out
when
$ml\_tl = \mathbf{green}$
then
$a := a + 1$
end

(concrete-)IL_out
when
$il\_tl = \mathbf{green}$
then
$b := b - 1$
$c := c + 1$
end

(concrete-)ML_in
when
$0 < c$
then
$c := c - 1$
end

(concrete-)IL_in
when
$0 < a$
then
$a := a - 1$
$b := b + 1$
end

- INV applied to ML_out and IL_out raise some difficulties

What we Have to Prove

- 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

$$\frac{H, P, Q \;\vdash\; R}{H,\; P,\; P \Rightarrow Q \;\vdash\; R} \quad \text{IMP\_L}$$

$$\frac{H, P \;\vdash\; Q}{H \;\vdash\; P \Rightarrow Q} \quad \text{IMP\_R}$$

- Rules about negation

$$\frac{H \;\vdash\; P}{H,\; \neg P \;\vdash\; Q} \quad \text{NOT\_L}$$

$$\frac{H, P \;\vdash\; Q \qquad H, P \;\vdash\; \neg Q}{H \;\vdash\; \neg P} \quad \text{NOT\_R}$$

Proving Preservation of inv2_4 by Event ML_out

axm0_1	$d \in \mathbb{N}$
axm0_2	$0 < d$
axm2_1	$COLOR = \{\text{green}, \text{red}\}$
axm2_2	$\text{green} \neq \text{red}$
inv0_1	$n \in \mathbb{N}$
inv0_2	$n \leq d$
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	$a = 0 \;\lor\; c = 0$
inv2_1	$ml\_tl \in COLOR$
inv2_2	$il\_tl \in COLOR$
inv2_3	$ml\_tl = \text{green} \Rightarrow a + b < d \;\land\; c = 0$
inv2_4	$il\_tl = \text{green} \Rightarrow 0 < b \;\land\; a = 0$
Guard of event ML_out	$ml\_tl = \text{green}$
$\vdash$	$\vdash$
Modified invariant **inv2_4**	$il\_tl = \text{green} \Rightarrow 0 < b \;\land\; a + 1 = 0$

ML_out / **inv2_4** / INV

ML_out
  when
    $ml\_tl = \text{green}$
  then
    $a := a + 1$
  end

Tentative Proof

$$
\begin{array}{l}
d \in \mathbb{N} \\
0 < d \\
COLOR = \{\mathsf{green}, \mathsf{red}\} \\
\mathsf{green} \neq \mathsf{red} \\
n \in \mathbb{N} \\
n \leq d \\
a \in \mathbb{N} \\
b \in \mathbb{N} \\
c \in \mathbb{N} \\
a + b + c = n \\
a = 0 \;\lor\; c = 0 \\
ml\_tl \in COLOR \\
il\_tl \in COLOR \\
ml\_tl = \mathsf{green} \;\Rightarrow\; a + b < d \;\land\; c = 0 \\
il\_tl = \mathsf{green} \;\Rightarrow\; 0 < b \;\land\; a = 0 \\
ml\_tl = \mathsf{green} \\
\vdash \\
il\_tl = \mathsf{green} \;\Rightarrow\; 0 < b \;\land\; a + 1 = 0
\end{array}
$$

MON

$$
\begin{array}{l}
\mathsf{green} \neq \mathsf{red} \\
il\_tl = \mathsf{green} \;\Rightarrow\; 0 < b \;\land\; a = 0 \\
ml\_tl = \mathsf{green} \\
\vdash \\
il\_tl = \mathsf{green} \;\Rightarrow\; 0 < b \;\land\; a + 1 = 0
\end{array}
$$

IMP_R $\cdots$

Tentative Proof

$$
\begin{array}{l}
d \in \mathbb{N} \\
0 < d \\
COLOR = \{\mathsf{green}, \mathsf{red}\} \\
\mathsf{green} \neq \mathsf{red} \\
n \in \mathbb{N} \\
n \leq d \\
a \in \mathbb{N} \\
b \in \mathbb{N} \\
c \in \mathbb{N} \\
a + b + c = n \\
a = 0 \ \vee \ c = 0 \\
ml\_tl \in COLOR \\
il\_tl \in COLOR \\
ml\_tl = \mathsf{green} \ \Rightarrow \ a + b < d \ \wedge \ c = 0 \\
il\_tl = \mathsf{green} \ \Rightarrow \ 0 < b \ \wedge \ a = 0 \\
ml\_tl = \mathsf{green} \\
\vdash \\
il\_tl = \mathsf{green} \ \Rightarrow \ 0 < b \ \wedge \ a + 1 = 0
\end{array}
$$

MON

$$
\begin{array}{l}
\mathsf{green} \neq \mathsf{red} \\
il\_tl = \mathsf{green} \ \Rightarrow \ 0 < b \ \wedge \ a = 0 \\
ml\_tl = \mathsf{green} \\
\vdash \\
il\_tl = \mathsf{green} \ \Rightarrow \ 0 < b \ \wedge \ a + 1 = 0
\end{array}
$$

IMP_R ⋯

⋯

$$
\begin{array}{l}
\mathsf{green} \neq \mathsf{red} \\
il\_tl = \mathsf{green} \ \Rightarrow \ 0 < b \ \wedge \ a = 0 \\
ml\_tl = \mathsf{green} \\
il\_tl = \mathsf{green} \\
\vdash \\
0 < b \ \wedge \ a + 1 = 0
\end{array}
$$

IMP_L

$$
\begin{array}{l}
\mathsf{green} \neq \mathsf{red} \\
0 < b \ \wedge \ a = 0 \\
ml\_tl = \mathsf{green} \\
il\_tl = \mathsf{green} \\
\vdash \\
0 < b \ \wedge \ a + 1 = 0
\end{array}
$$

AND_L ⋯

Tentative Proof (cont'd)

$\ldots$

$\mathsf{green} \neq \mathsf{red}$
$0 < b$
$a = 0$
$ml\_tl = \mathsf{green}$
$il\_tl = \mathsf{green}$
$\vdash$
$0 < b \;\wedge\; a + 1 = 0$

AND_R

$\mathsf{green} \neq \mathsf{red}$
$0 < b$
$a = 0$
$ml\_tl = \mathsf{green}$
$il\_tl = \mathsf{green}$
$\vdash$
$0 < b$

MON

$0 < b \;\vdash\; 0 < b$

HYP

$\mathsf{green} \neq \mathsf{red}$
$0 < b$
$a = 0$
$ml\_tl = \mathsf{green}$
$il\_tl = \mathsf{green}$
$\vdash$
$a + 1 = 0$

EQ_LR

$\mathsf{green} \neq \mathsf{red}$
$ml\_tl = \mathsf{green}$
$il\_tl = \mathsf{green}$
$\vdash$
$\underline{0 + 1 = 0}$

ARITH

$\mathsf{green} \neq \mathsf{red}$
$ml\_tl = \mathsf{green}$
$il\_tl = \mathsf{green}$
$\vdash$
$1 = 0$

?

Proving Preservation of inv2_3 by Event IL_out

axm0_1	$d \in \mathbb{N}$
axm0_2	$0 < d$
axm2_1	$COLOR = \{\text{green}, \text{red}\}$
axm2_2	$\text{green} \neq \text{red}$
inv0_1	$n \in \mathbb{N}$
inv0_2	$n \leq d$
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	$a = 0 \ \lor \ c = 0$
inv2_1	$ml\_tl \in COLOR$
inv2_2	$il\_tl \in COLOR$
inv2_3	$ml\_tl = \text{green} \ \Rightarrow \ a + b < d \ \land \ c = 0$
inv2_4	$il\_tl = \text{green} \ \Rightarrow \ 0 < b \ \land \ a = 0$
Guard of IL_out	$il\_tl = \text{green}$
$\vdash$	$\vdash$
Modified **inv2_3**	$ml\_tl = \text{green} \ \Rightarrow \ a + b - 1 < d \ \land \ c + 1 = 0$

IL_out / **inv2_3** / INV

IL_out
when
$il\_tl = \text{green}$
then
$b := b - 1$
$c := c + 1$
end

Tentative Proof

$$
\begin{array}{l}
d \in \mathbb{N} \\
0 < d \\
COLOR = \{\textbf{green}, \textbf{red}\} \\
\textbf{green} \neq \textbf{red} \\
n \in \mathbb{N} \\
n \leq d \\
a \in \mathbb{N} \\
b \in \mathbb{N} \\
c \in \mathbb{N} \\
a + b + c = n \\
a = 0 \;\vee\; c = 0 \\
ml\_tl \in COLOR \\
il\_tl \in COLOR \\
ml\_tl = \textbf{green} \Rightarrow a + b < d \wedge c = 0 \\
il\_tl = \textbf{green} \Rightarrow 0 < b \wedge a = 0 \\
il\_tl = \textbf{green} \\
\vdash \\
ml\_tl = \textbf{green} \Rightarrow a + b - 1 < d \wedge c + 1 = 0
\end{array}
$$

MON

$$
\begin{array}{l}
\textbf{green} \neq \textbf{red} \\
ml\_tl = \textbf{green} \Rightarrow a + b < d \wedge c = 0 \\
il\_tl = \textbf{green} \\
\vdash \\
ml\_tl = \textbf{green} \Rightarrow a + b - 1 < d \wedge c + 1 = 0
\end{array}
$$

IMP_R · ·

Tentative Proof

$$
\begin{array}{l}
d \in \mathbb{N} \\
0 < d \\
COLOR = \{\textsf{green}, \textsf{red}\} \\
\textsf{green} \neq \textsf{red} \\
n \in \mathbb{N} \\
n \leq d \\
a \in \mathbb{N} \\
b \in \mathbb{N} \\
c \in \mathbb{N} \\
a + b + c = n \\
a = 0 \;\vee\; c = 0 \\
ml\_tl \in COLOR \\
il\_tl \in COLOR \\
ml\_tl = \textsf{green} \Rightarrow a + b < d \wedge c = 0 \\
il\_tl = \textsf{green} \Rightarrow 0 < b \wedge a = 0 \\
il\_tl = \textsf{green} \\
\vdash \\
ml\_tl = \textsf{green} \Rightarrow a + b - 1 < d \wedge c + 1 = 0
\end{array}
$$

MON

$$
\begin{array}{l}
\textsf{green} \neq \textsf{red} \\
ml\_tl = \textsf{green} \Rightarrow a + b < d \wedge c = 0 \\
il\_tl = \textsf{green} \\
\vdash \\
ml\_tl = \textsf{green} \Rightarrow a + b - 1 < d \;\wedge \\
\qquad c + 1 = 0
\end{array}
$$

IMP_R ···

···

$$
\begin{array}{l}
\textsf{green} \neq \textsf{red} \\
ml\_tl = \textsf{green} \Rightarrow a + b < d \wedge c = 0 \\
il\_tl = \textsf{green} \\
ml\_tl = \textsf{green} \\
\vdash \\
a + b - 1 < d \wedge c + 1 = 0
\end{array}
$$

IMP_L

$$
\begin{array}{l}
\textsf{green} \neq \textsf{red} \\
a + b < d \wedge c = 0 \\
il\_tl = \textsf{green} \\
ml\_tl = \textsf{green} \\
\vdash \\
a + b - 1 < d \;\wedge \\
c + 1 = 0
\end{array}
$$

AND_L ···

Tentative Proof (cont'd)

. . .

green $\neq$ red
$a + b < d$
$c = 0$
$il\_tl =$ green
$ml\_tl =$ green
$\vdash$
$a + b - 1 < d \ \wedge$
$c + 1 = 0$

AND_R

green $\neq$ red
$a + b < d$
$c = 0$
$il\_tl =$ green
$ml\_tl =$ green
$\vdash$
$a + b - 1 < d$

MON

$a + b < d \ \vdash \ a + b - 1 < d$

DEC

green $\neq$ red
$c = 0$
$il\_tl =$ green
$ml\_tl =$ green
$\vdash$
$c + 1 = 0$

EQ_LR

green $\neq$ red
$il\_tl =$ green
$ml\_tl =$ green
$\vdash$
$\underline{0 + 1 = 0}$

ARITH

green $\neq$ red
$il\_tl =$ green
$ml\_tl =$ green
$\vdash$
$1 = 0$

?

The Solution

- In both cases, we were stopped by attempting to prove the following

$$\begin{array}{l} \mathbf{green} \neq \mathbf{red} \\ il\_tl = \mathbf{green} \\ ml\_tl = \mathbf{green} \\ \vdash \\ 1 = 0 \end{array}$$

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

inv2_5: $ml\_tl = \textbf{red} \quad \lor \quad il\_tl = \textbf{red}$

Completing the Proof

$$\begin{array}{l} \textsf{green} \neq \textsf{red} \\ ml\_tl = \textsf{red} \;\lor\; il\_tl = \textsf{red} \\ il\_tl = \textsf{green} \\ ml\_tl = \textsf{green} \\ \vdash \\ 1 = 0 \end{array}$$

OR_L

$$\begin{array}{l} \textsf{green} \neq \textsf{red} \\ ml\_tl = \textsf{red} \\ il\_tl = \textsf{green} \\ ml\_tl = \textsf{green} \\ \vdash \\ 1 = 0 \end{array} \quad \textsf{EQ\_LR} \quad \begin{array}{l} \textsf{green} \neq \textsf{red} \\ \textsf{green} = \textsf{red} \\ il\_tl = \textsf{green} \\ \vdash \\ 1 = 0 \end{array} \quad \textsf{NOT\_L} \cdots$$

$$\begin{array}{l} \textsf{green} \neq \textsf{red} \\ il\_tl = \textsf{red} \\ il\_tl = \textsf{green} \\ ml\_tl = \textsf{green} \\ \vdash \\ 1 = 0 \end{array} \quad \textsf{EQ\_LR} \quad \begin{array}{l} \textsf{green} \neq \textsf{red} \\ \textsf{green} = \textsf{red} \\ ml\_tl = \textsf{green} \\ \vdash \\ 1 = 0 \end{array} \quad \textsf{NOT\_L} \cdots$$

$$\cdots \quad \begin{array}{l} \textsf{green} = \textsf{red} \\ il\_tl = \textsf{green} \\ \vdash \\ \textsf{green} = \textsf{red} \end{array} \quad \textsf{MON} \quad \textsf{green} = \textsf{red} \;\vdash\; \textsf{green} = \textsf{red} \quad \textsf{HYP}$$

$$\cdots \quad \begin{array}{l} \textsf{green} = \textsf{red} \\ ml\_tl = \textsf{green} \\ \vdash \\ \textsf{green} = \textsf{red} \end{array} \quad \textsf{MON} \quad \textsf{green} = \textsf{red} \;\vdash\; \textsf{green} = \textsf{red} \quad \textsf{HYP}$$

Going back to the Requirements Document

inv2_5: $ml\_tl = \textbf{red} \quad \vee \quad il\_tl = \textbf{red}$

This could have been deduced from these requirements

The bridge is one way or the other, not both at the same time	FUN-3

Cars are not supposed to pass on a red traffic light, only on a green one	EQP-3

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

axm0_1	$d \in \mathbb{N}$
axm0_2	$0 < d$
axm2_1	$COLOR = \{\text{green}, \text{red}\}$
axm2_2	$\text{green} \neq \text{red}$
inv0_1	$n \in \mathbb{N}$
inv0_2	$n \leq d$
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	$a = 0 \lor c = 0$
inv2_1	$ml\_tl \in COLOR$
inv2_2	$il\_tl \in COLOR$
inv2_3	$ml\_tl = \text{green} \Rightarrow a + b < d \land c = 0$
inv2_4	$il\_tl = \text{green} \Rightarrow 0 < b \land a = 0$
Guard of ML_out	$ml\_tl = \text{green}$
$\vdash$	$\vdash$
Modified **inv2_3**	$ml\_tl = \text{green} \Rightarrow a + 1 + b < d \land c = 0$

ML_out / inv2_3 / **INV**

ML_out
when
$ml\_tl = \text{green}$
then
$a := a + 1$
end

Tentative Proof

$$
\begin{array}{l}
d \in \mathbb{N} \\
0 < d \\
COLOR = \{\textbf{green}, \textbf{red}\} \\
\textbf{green} \neq \textbf{red} \\
n \in \mathbb{N} \\
n \leq d \\
a \in \mathbb{N} \\
b \in \mathbb{N} \\
c \in \mathbb{N} \\
a + b + c = n \\
a = 0 \;\vee\; c = 0 \\
ml\_tl \in COLOR \\
il\_tl \in COLOR \\
ml\_tl = \textbf{green} \Rightarrow a + b < d \;\wedge\; c = 0 \\
il\_tl = \textbf{green} \Rightarrow 0 < b \;\wedge\; a = 0 \\
ml\_tl = \textbf{green} \\
\vdash \\
ml\_tl = \textbf{green} \Rightarrow a + 1 + b < d \;\wedge\; c = 0
\end{array}
$$

MON

$$
\begin{array}{l}
ml\_tl = \textbf{green} \Rightarrow a + b < d \;\wedge\; c = 0 \\
\vdash \\
ml\_tl = \textbf{green} \Rightarrow a + 1 + b < d \;\wedge\; c = 0
\end{array}
$$

IMP_R $\cdots$

Tentative Proof

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

MON

$ml\_tl = \text{green} \Rightarrow a + b < d \land c = 0$
$\vdash$
$ml\_tl = \text{green} \Rightarrow a + 1 + b < d \land c = 0$

IMP_R ...

...

$ml\_tl = \text{green} \Rightarrow a + b < d \land c = 0$
$ml\_tl = \text{green}$
$\vdash$
$a + 1 + b < d \land c = 0$

IMP_L

$a + b < d \land c = 0$
$ml\_tl = \text{green}$
$\vdash$
$a + 1 + b < d \land c = 0$

AND_L ...

Tentative Proof (cont'd)

$\ldots$

$$a + b < d,\quad c = 0,\quad ml\_tl = \textsf{green} \;\vdash\; a + 1 + b < d \;\wedge\; c = 0$$

AND_R

$$a + b < d,\quad c = 0,\quad ml\_tl = \textsf{green} \;\vdash\; a + 1 + b < d$$

MON

$$a + b < d,\quad ml\_tl = \textsf{green} \;\vdash\; a + 1 + b < d$$

?

$$a + b < d,\quad c = 0,\quad ml\_tl = \textsf{green} \;\vdash\; c = 0$$

MON

$$c = 0 \;\vdash\; c = 0$$

HYP

- This requires splitting the ML_out in two separate events ML_out_1 and ML_out_2

```
ML_out_1
  when
    ml_tl = green
    a + 1 + b < d
  then
    a := a + 1
  end
```

```
ML_out_2
  when
    ml_tl = green
    a + 1 + b = d
  then
    a := a + 1
    ml_tl := red
  end
```

Intuitive Explanation

ML_out_1
  when
    $ml\_tl = \textbf{green}$
    $a + 1 + b < d$
  then
    $a := a + 1$
  end

ML_out_2
  when
    $ml\_tl = \textbf{green}$
    $a + 1 + b = d$
  then
    $a := a + 1$
    $ml\_tl := \textbf{red}$
  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 inv2_3 by Event ML_out_1

axm0_1	$d \in \mathbb{N}$
axm0_2	$0 < d$
axm2_1	$COLOR = \{\text{green}, \text{red}\}$
axm2_2	$\text{green} \neq \text{red}$
inv0_1	$n \in \mathbb{N}$
inv0_2	$n \leq d$
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	$a = 0 \lor c = 0$
inv2_1	$ml\_tl \in COLOR$
inv2_2	$il\_tl \in COLOR$
inv2_3	$ml\_tl = \text{green} \Rightarrow a + b < d \land c = 0$
inv2_4	$il\_tl = \text{green} \Rightarrow 0 < b \land a = 0$
Guard of ML_out_1	$ml\_tl = \text{green}$
	$a + 1 + b < d$
$\vdash$	$\vdash$
Modified **inv2_3**	$ml\_tl = \text{green} \Rightarrow a + 1 + b < d \land c = 0$

ML_out_1 / inv2_3 / INV

ML_out_1
when
$ml\_tl = \text{green}$
$a + 1 + b < d$
then
$a := a + 1$
end

Proof

$$
\begin{array}{l}
d \in \mathbb{N} \\
0 < d \\
COLOR = \{\textbf{green}, \textbf{red}\} \\
\textbf{green} \neq \textbf{red} \\
n \in \mathbb{N} \\
n \leq d \\
a \in \mathbb{N} \\
b \in \mathbb{N} \\
c \in \mathbb{N} \\
a + b + c = n \\
a = 0 \;\lor\; c = 0 \\
ml\_tl \in COLOR \\
il\_tl \in COLOR \\
ml\_tl = \textbf{green} \;\Rightarrow\; a + b < d \;\land\; c = 0 \\
il\_tl = \textbf{green} \;\Rightarrow\; 0 < b \;\land\; a = 0 \\
ml\_tl = \textbf{green} \\
a + 1 + b < d \\
\vdash \\
ml\_tl = \textbf{green} \;\Rightarrow\; a + 1 + b < d \;\land\; c = 0
\end{array}
$$

MON

$$
\begin{array}{l}
ml\_tl = \textbf{green} \;\Rightarrow\; a + b < d \;\land\; c = 0 \\
a + 1 + b < d \\
\vdash \\
ml\_tl = \textbf{green} \;\Rightarrow\; a + 1 + b < d \;\land\; c = 0
\end{array}
$$

IMP_R $\cdots$

Proof

$$
\begin{array}{l}
d \in \mathbb{N} \\
0 < d \\
COLOR = \{\mathbf{green}, \mathbf{red}\} \\
\mathbf{green} \neq \mathbf{red} \\
n \in \mathbb{N} \\
n \leq d \\
a \in \mathbb{N} \\
b \in \mathbb{N} \\
c \in \mathbb{N} \\
a + b + c = n \\
a = 0 \;\vee\; c = 0 \\
ml\_tl \in COLOR \\
il\_tl \in COLOR \\
ml\_tl = \mathbf{green} \;\Rightarrow\; a + b < d \;\wedge\; c = 0 \\
il\_tl = \mathbf{green} \;\Rightarrow\; 0 < b \;\wedge\; a = 0 \\
ml\_tl = \mathbf{green} \\
a + 1 + b < d \\
\vdash \\
ml\_tl = \mathbf{green} \;\Rightarrow\; a + 1 + b < d \;\wedge\; c = 0
\end{array}
$$

MON

$$
\begin{array}{l}
ml\_tl = \mathbf{green} \;\Rightarrow\; a + b < d \;\wedge\; c = 0 \\
a + 1 + b < d \\
\vdash \\
ml\_tl = \mathbf{green} \;\Rightarrow\; a + 1 + b < d \;\wedge\; c = 0
\end{array}
$$

IMP_R $\cdots$

$\cdots$

$$
\begin{array}{l}
ml\_tl = \mathbf{green} \;\Rightarrow\; a + b < d \;\wedge\; c = 0 \\
ml\_tl = \mathbf{green} \\
a + 1 + b < d \\
\vdash \\
a + 1 + b < d \;\wedge\; c = 0
\end{array}
$$

IMP_L

$$
\begin{array}{l}
a + b < d \;\wedge\; c = 0 \\
ml\_tl = \mathbf{green} \\
a + 1 + b < d \\
\vdash \\
a + 1 + b < d \;\wedge\; c = 0
\end{array}
$$

AND_L $\cdots$

Proof (cont'd)

$\ldots$

$$\begin{array}{l} a+b<d \\ c=0 \\ ml\_tl = \textbf{green} \\ a+1+b<d \\ \vdash \\ a+1+b<d \quad \wedge \quad c=0 \end{array}$$

AND_R

$$\begin{array}{l} a+b<d \\ c=0 \\ ml\_tl = \textbf{green} \\ a+1+b<d \\ \vdash \\ a+1+b<d \end{array}$$

MON

$$\begin{array}{l} a+1+b<d \\ \vdash \\ a+1+b<d \end{array}$$

HYP

$$\begin{array}{l} a+b<d \\ c=0 \\ ml\_tl = \textbf{green} \\ a+1+b<d \\ \vdash \\ c=0 \end{array}$$

MON

$$c=0 \quad \vdash \quad c=0$$

HYP

Proving Preservation of inv2_3 by Event ML_out_2

axm0_1	$d \in \mathbb{N}$	
axm0_2	$0 < d$	
axm2_1	$COLOR = \{\text{green}, \text{red}\}$	
axm2_2	$\text{green} \neq \text{red}$	
inv0_1	$n \in \mathbb{N}$	
inv0_2	$n \leq d$	
inv1_1	$a \in \mathbb{N}$	
inv1_2	$b \in \mathbb{N}$	
inv1_3	$c \in \mathbb{N}$	
inv1_4	$a + b + c = n$	ML_out_2 / inv2_3 / INV
inv1_5	$a = 0 \;\vee\; c = 0$	
inv2_1	$ml\_tl \in COLOR$	
inv2_2	$il\_tl \in COLOR$	
inv2_3	$ml\_tl = \text{green} \Rightarrow a + b < d \wedge c = 0$	
inv2_4	$il\_tl = \text{green} \Rightarrow 0 < b \wedge a = 0$	
Guard of ML_out_2	$ml\_tl = \text{green}$	
	$a + 1 + b = d$	
$\vdash$	$\vdash$	
Modified **inv2_3**	$\text{red} = \text{green} \Rightarrow a + 1 + b < d \wedge c = 0$	

ML_out_2
  when
    $ml\_tl = \text{green}$
    $a + 1 + b = d$
  then
    $a := a + 1$
    $ml\_tl := red$
  end

Proof

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

MON

$\text{green} \neq \text{red}$
$\vdash$
$\text{red} = \text{green} \Rightarrow a + 1 + b < d \land c = 0$

IMP_R

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

MON

$\text{green} \neq \text{red}$
$\vdash$
$\text{red} = \text{green} \Rightarrow a + 1 + b < d \land c = 0$

IMP_R

...

$\text{green} \neq \text{red}$
$\text{red} = \text{green}$
$\vdash$
$a + 1 + b < d \land c = 0$

EQ_LR

$\text{green} \neq \text{green}$
$\vdash$
$a + 1 + b < d \land c = 0$

NOT_L

$\vdash$
$\text{green} = \text{green}$

EQL

What we Have to Prove

- ML_out / **inv2_4** / INV done

- IL_out / **inv2_3** / INV done

- ML_out / **inv2_3** / INV done

- IL_out / **inv2_4** / INV

- ML_tl_green / **inv2_5** / INV

- IL_tl_green / **inv2_5** / INV

axm0_1	$d \in \mathbb{N}$
axm0_2	$0 < d$
axm2_1	$COLOR = \{\text{green}, \text{red}\}$
axm2_2	$\text{green} \neq \text{red}$
inv0_1	$n \in \mathbb{N}$
inv0_2	$n \leq d$
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	$a = 0 \ \vee \ c = 0$
inv2_1	$ml\_tl \in COLOR$
inv2_2	$il\_tl \in COLOR$
inv2_3	$ml\_tl = \text{green} \Rightarrow a + b < d \ \wedge \ c = 0$
inv2_4	$il\_tl = \text{green} \Rightarrow 0 < b \ \wedge \ a = 0$
Guard of event IL_out	$il\_tl = \text{green}$
$\vdash$	$\vdash$
Modified invariant **inv2_4**	$il\_tl = \text{green} \Rightarrow 0 < b - 1 \ \wedge \ a = 0$

IL_out / **inv2_4** / INV

IL_out
  when
    $il\_tl = \text{green}$
  then
    $b := b - 1$
    $c := c + 1$
  end

Tentative Proof

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

MON

$il\_tl = \text{green} \Rightarrow 0 < b \land a = 0$
$il\_tl = \text{green}$
$\vdash$
$il\_tl = \text{green} \Rightarrow 0 < b - 1 \land a = 0$

IMP_R

...

$il\_tl = \text{green} \Rightarrow 0 < b \land a = 0$
$il\_tl = \text{green}$
$\vdash$
$0 < b - 1 \land a = 0$

IMP_L

$0 < b \land a = 0$
$\vdash$
$0 < b - 1 \land a = 0$

AND_L

Tentative Proof (cont’d)

$$\ldots \quad \begin{array}{l} 0 < b \\ a = 0 \\ \vdash \\ 0 < b - 1 \;\wedge\; a = 0 \end{array} \quad \text{AND\_R} \left\{ \begin{array}{l} \begin{array}{l} 0 < b \\ a = 0 \\ \vdash \\ 0 < b - 1 \end{array} \quad \text{MON} \quad \begin{array}{l} 0 < b \\ \vdash \\ 0 < b - 1 \end{array} \quad ? \\ \\ \begin{array}{l} 0 < b \\ a = 0 \\ \vdash \\ a = 0 \end{array} \quad \text{MON} \quad \begin{array}{l} a = 0 \\ \vdash \\ a = 0 \end{array} \quad \text{HYP} \end{array} \right.$$

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

IL_out_1
when
$il\_tl = \text{green}$
$b \neq 1$
then
$b, c := b - 1, c + 1$
end

IL_out_2
when
$il\_tl = \text{green}$
$b = 1$
then
$b, c := b - 1, c + 1$
$il\_tl := \text{red}$
end

Intuitive Explanation


```
IL_out_1
   when
      il_tl = green
      b ≠ 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)

axm0_1	$d \in \mathbb{N}$
axm0_2	$0 < d$
axm2_1	$COLOR = \{\text{green}, \text{red}\}$
axm2_2	$\text{green} \neq \text{red}$
inv0_1	$n \in \mathbb{N}$
inv0_2	$n \leq d$
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	$a = 0 \;\vee\; c = 0$
inv2_1	$ml\_tl \in COLOR$
inv2_2	$il\_tl \in COLOR$
inv2_3	$ml\_tl = \text{green} \;\Rightarrow\; a + b < d \;\wedge\; c = 0$
inv2_4	$il\_tl = \text{green} \;\Rightarrow\; 0 < b \;\wedge\; a = 0$
Guard of event IL_out_1	$il\_tl = \text{green}$ $b \neq 1$
$\vdash$	$\vdash$
Modified invariant **inv2_4**	$il\_tl = \text{green} \;\Rightarrow\; 0 < b - 1 \;\wedge\; a = 0$

IL_out_1 / **inv2_4** / INV

IL_out_1
  when
    $il\_tl = \text{green}$
    $b \neq 1$
  then
    $b, c := b - 1, c + 1$
  end

Proof

$$
\begin{array}{l}
d \in \mathbb{N} \\
0 < d \\
COLOR = \{\text{green}, \text{red}\} \\
\text{green} \neq \text{red} \\
n \in \mathbb{N} \\
n \leq d \\
a \in \mathbb{N} \\
b \in \mathbb{N} \\
c \in \mathbb{N} \\
a + b + c = n \\
a = 0 \ \vee \ c = 0 \\
ml\_tl \in COLOR \\
il\_tl \in COLOR \\
ml\_tl = \text{green} \Rightarrow a + b < d \ \wedge \ c = 0 \\
il\_tl = \text{green} \Rightarrow 0 < b \ \wedge \ a = 0 \\
il\_tl = \text{green} \\
b \neq 1 \\
\vdash \\
il\_tl = \text{green} \Rightarrow 0 < b - 1 \ \wedge \ a = 0
\end{array}
$$

MON

$$
\begin{array}{l}
il\_tl = \text{green} \Rightarrow 0 < b \ \wedge \ a = 0 \\
il\_tl = \text{green} \\
b \neq 1 \\
\vdash \\
il\_tl = \text{green} \Rightarrow 0 < b - 1 \ \wedge \ a = 0
\end{array}
$$

IMP_R

...

$$
\begin{array}{l}
il\_tl = \text{green} \Rightarrow 0 < b \ \wedge \ a = 0 \\
il\_tl = \text{green} \\
b \neq 1 \\
\vdash \\
0 < b - 1 \ \wedge \ a = 0
\end{array}
$$

IMP_L

$$
\begin{array}{l}
0 < b \ \wedge \ a = 0 \\
b \neq 1 \\
\vdash \\
0 < b - 1 \ \wedge \ a = 0
\end{array}
$$

AND_L

Proof (cont'd)

$$\ldots \quad \left[\begin{array}{l} 0 < b \\ a = 0 \\ b \neq 1 \\ \vdash \\ 0 < b - 1 \ \wedge \ a = 0 \end{array}\right] \text{ AND\_R}$$

$$\left[\begin{array}{l} 0 < b \\ a = 0 \\ b \neq 1 \\ \vdash \\ 0 < b - 1 \end{array}\right] \text{ MON } \left[\begin{array}{l} 0 < b \\ b \neq 1 \\ \vdash \\ 0 < b - 1 \end{array}\right] \text{ ARITH } \left[\begin{array}{l} 0 < b - 1 \\ \vdash \\ 0 < b - 1 \end{array}\right] \text{ HYP}$$

$$\left[\begin{array}{l} 0 < b \\ a = 0 \\ b \neq 1 \quad \vdash \\ a = 0 \end{array}\right] \text{ MON } \left[\begin{array}{l} a = 0 \\ \vdash \\ a = 0 \end{array}\right] \text{ HYP}$$

Proving Preservation of inv2_4 by Event IL_out_2

axm0_1	$d \in \mathbb{N}$
axm0_2	$0 < d$
axm2_1	$COLOR = \{\text{green}, \text{red}\}$
axm2_2	$\text{green} \neq \text{red}$
inv0_1	$n \in \mathbb{N}$
inv0_2	$n \leq d$
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	$a = 0 \lor c = 0$
inv2_1	$ml\_tl \in COLOR$
inv2_2	$il\_tl \in COLOR$
inv2_3	$ml\_tl = \text{green} \Rightarrow a + b < d \land c = 0$
inv2_4	$il\_tl = \text{green} \Rightarrow 0 < b \land a = 0$
Guard of event IL_out_2	$il\_tl = \text{green}$
	$b = 1$
$\vdash$	$\vdash$
Modified invariant **inv2_4**	$\text{red} = \text{green} \Rightarrow 0 < b - 1 \land a = 0$

IL_out_1 / **inv2_4** / INV

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

Proof

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

MON

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

IMP_R

...

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

EQ_LR

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

NOT_L

$\vdash$
$\text{green} = \text{green}$

EQL

What we Have to Prove

- ML_out / **inv2_4** / INV done

- IL_out / **inv2_3** / INV done

- ML_out / **inv2_3** / INV done

- IL_out / **inv2_4** / INV done

- ML_tl_green / **inv2_5** / INV

- IL_tl_green / **inv2_5** / INV

Correcting the New Events

But the new invariant **inv2_5** is not preserved by the new events

inv2_5: $ml\_tl = \textbf{red} \;\vee\; il\_tl = \textbf{red}$

Unless we correct them as follows:

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

IL_tl_green
when
$il\_tl = \textbf{red}$
$0 < b$
$a = 0$
then
$il\_tl := \textbf{green}$
$ml\_tl := \textbf{red}$
end

Summary of the Proof Situation

- Correct event refinement: OK

- Absence of divergence of new events: FAILURE

- Absence of deadlock: ?

Divergence of the New Events

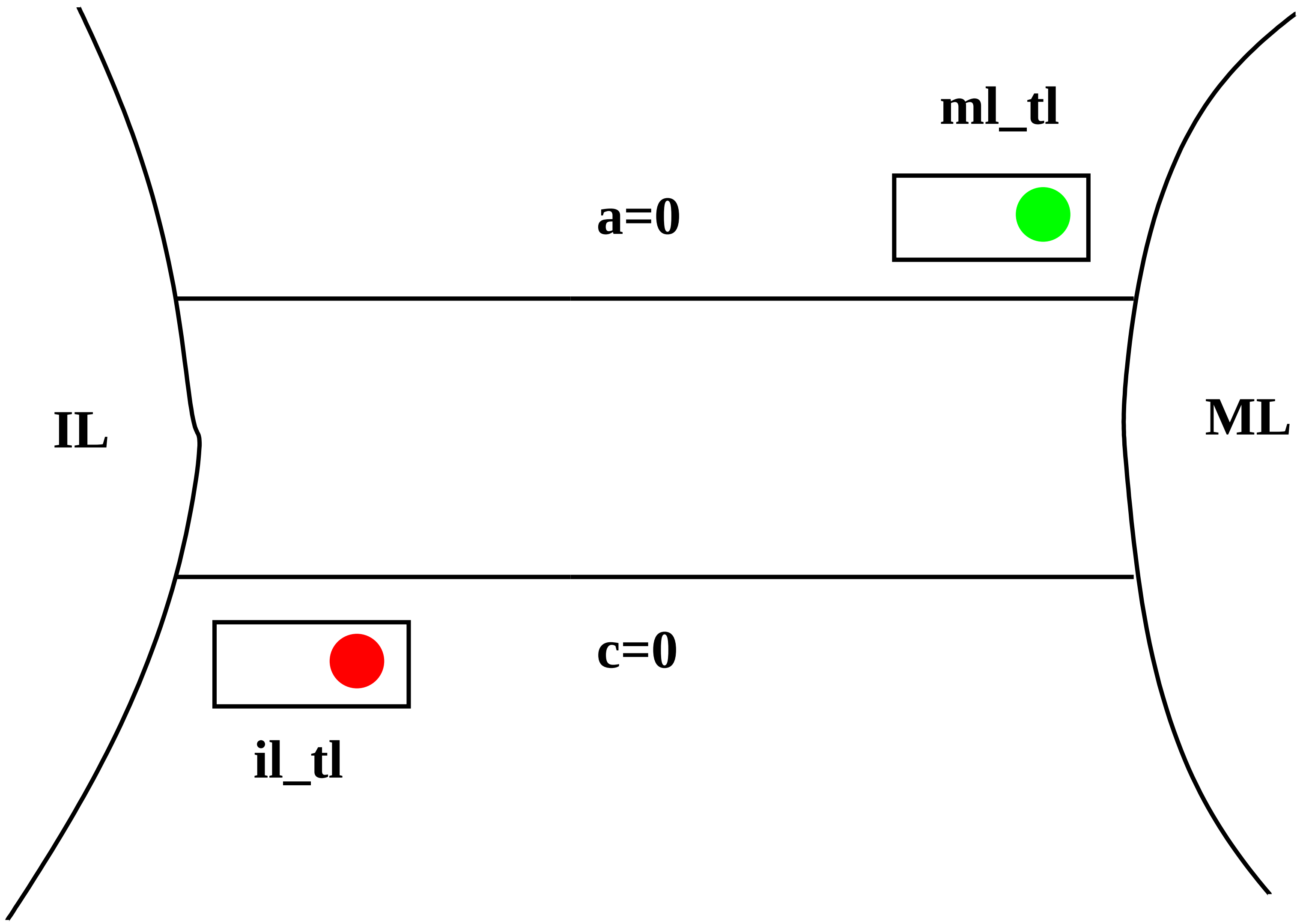
ML_tl_green
when
$ml\_tl = \textbf{red}$
$a + b < d$
$c = 0$
then
$ml\_tl := \textbf{green}$
$il\_tl := \textbf{red}$
end

IL_tl_green
when
$il\_tl = \textbf{red}$
$0 < b$
$a = 0$
then
$il\_tl := \textbf{green}$
$ml\_tl := \textbf{red}$
end

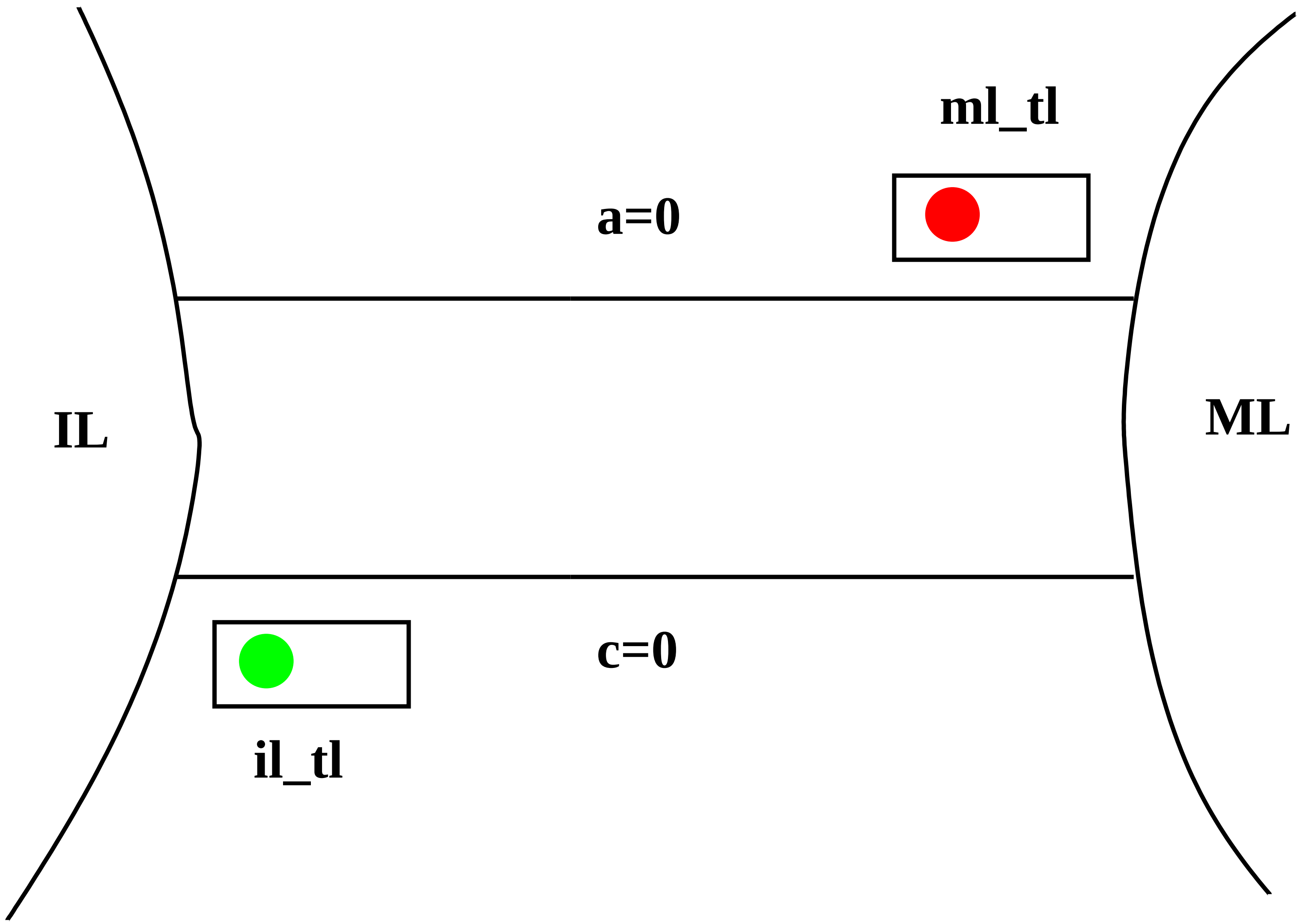
When a and c are both equal to 0 and b is positive, then both events are always alternatively enabled

The lights can change colors very rapidly

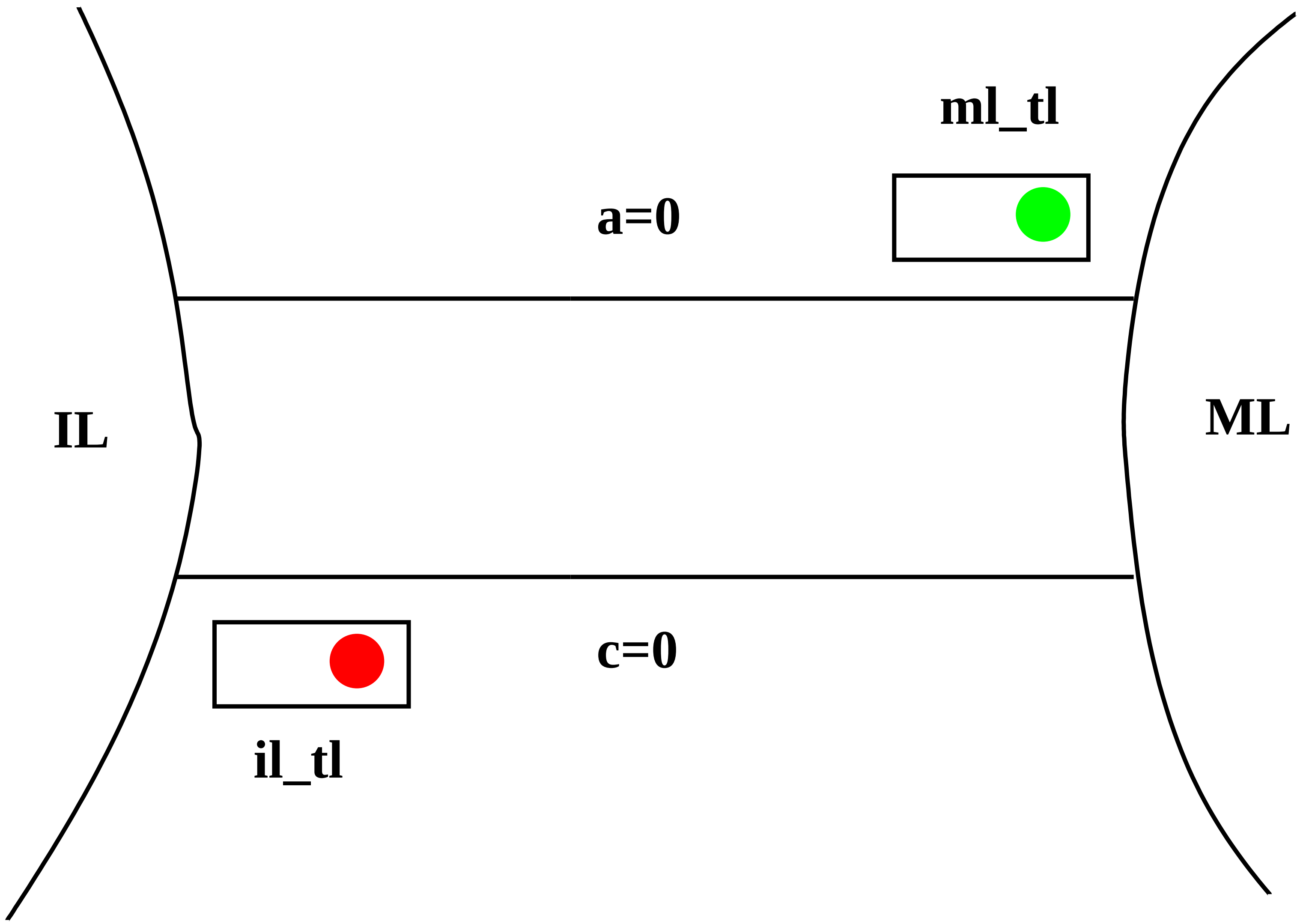
ML_tl_green and IL_tl_green can run for ever



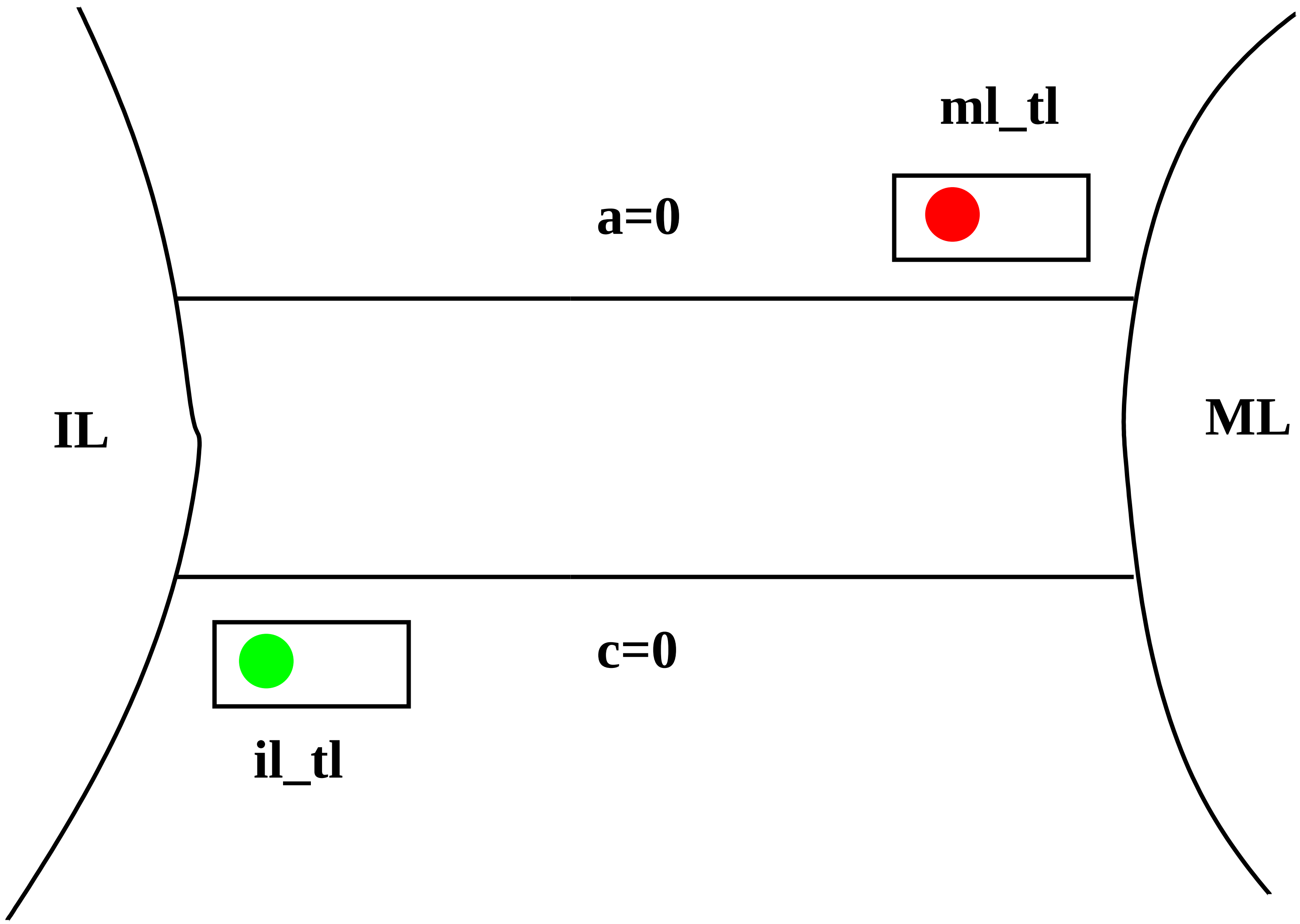
ML_tl_green and IL_tl_green can run for ever



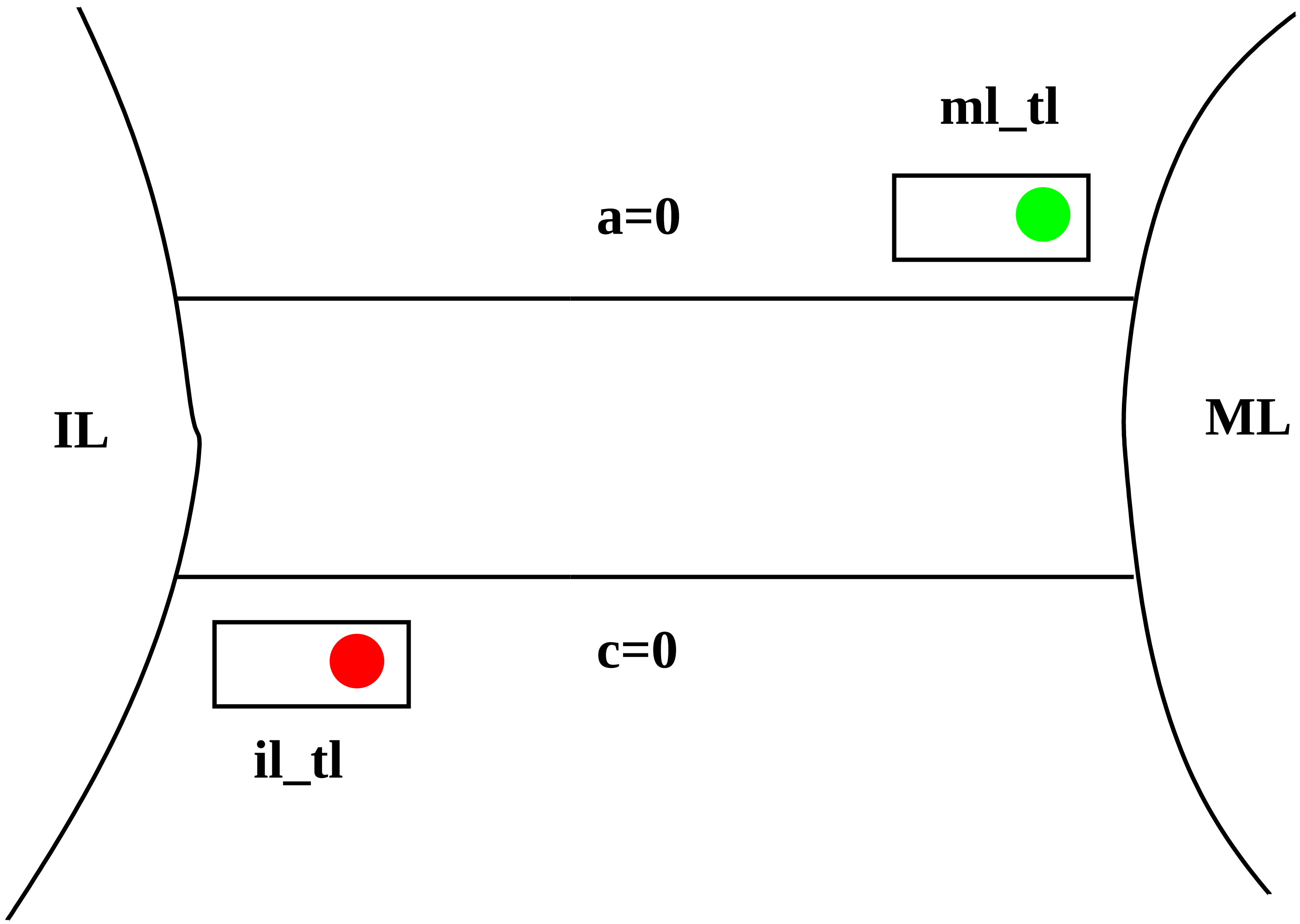
ML_tl_green and IL_tl_green can run for ever



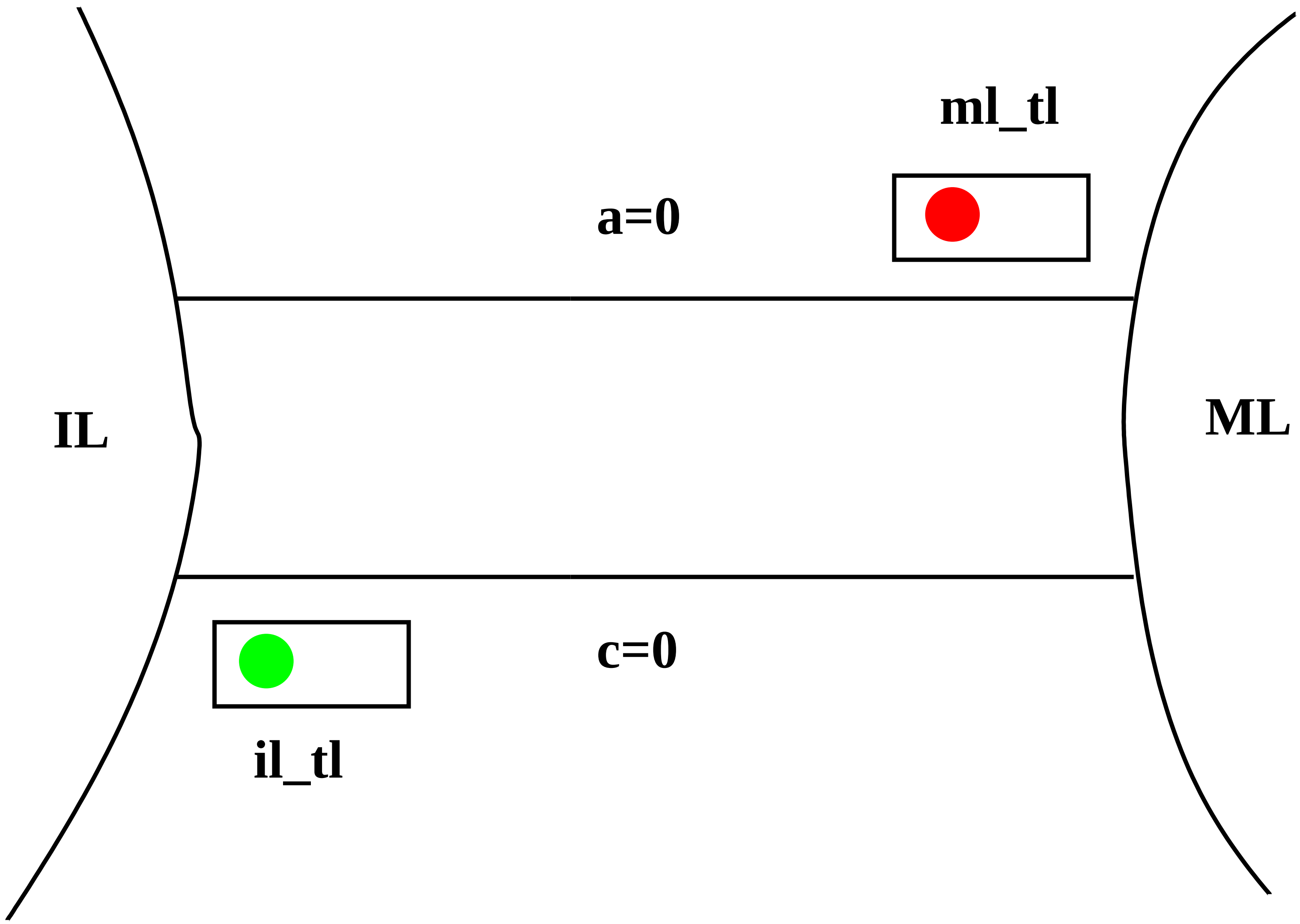
ML_tl_green and IL_tl_green can run for ever



ML_tl_green and IL_tl_green can run for ever



ML_tl_green and IL_tl_green can run for ever



Solution

- 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:

inv2_6: $ml\_pass \in \{0, 1\}$

inv2_7: $il\_pass \in \{0, 1\}$

Modifying Events ML_out_1 and ML_out_2

ML_out_1
  when
    $ml\_tl = \textbf{green}$
    $a + 1 + b < d$
  then
    $a := a + 1$
    $ml\_pass := 1$
  end

ML_out_2
  when
    $ml\_tl = \textbf{green}$
    $a + 1 + b = d$
  then
    $a := a + 1$
    $ml\_tl := \textbf{red}$
    $ml\_pass := 1$
  end

Modifying Events ML_out_1 and ML_out_2

IL_out_1
when
$il\_tl = \textbf{green}$
$b \neq 1$
then
$b := b - 1$
$c := c + 1$
$il\_pass := 1$
end

IL_out_2
when
$il\_tl = \textbf{green}$
$b = 1$
then
$b := b - 1$
$c := c + 1$
$il\_tl := \textbf{red}$
$il\_pass := 1$
end

Modifying Events ML_tl_gree and IL_tl_green

ML_tl_green
when
$ml\_tl = \textbf{red}$
$a + b < d$
$c = 0$
$il\_pass = 1$
then
$ml\_tl := \textbf{green}$
$il\_tl := \textbf{red}$
$ml\_pass := 0$
end

IL_tl_green
when
$il\_tl = \textbf{red}$
$0 < b$
$a = 0$
$ml\_pass = 1$
then
$il\_tl := \textbf{green}$
$ml\_tl := \textbf{red}$
$il\_pass := 0$
end

Proving Absence of Divergence

We exhibit the following variant

variant_2: $ml\_pass + il\_pass$

To be Proved

$$
\begin{array}{l}
\quad ml\_tl = \textbf{red} \\
\quad a + b < d \\
\quad c = 0 \\
\quad il\_pass = 1 \\
\Rightarrow \\
\quad il\_pass + 0 < \\
\quad ml\_pass + il\_pass
\end{array}
$$

$$
\begin{array}{l}
\quad il\_tl = \textbf{red} \\
\quad b > 0 \\
\quad a = 0 \\
\quad ml\_pass = 1 \\
\Rightarrow \\
\quad ml\_pass + 0 < \\
\quad ml\_pass + il\_pass
\end{array}
$$

This cannot be proved. This suggests the following invariants:

inv2_8: $ml\_tl = \textbf{red} \;\Rightarrow\; ml\_pass = 1$

inv2_9: $il\_tl = \textbf{red} \;\Rightarrow\; il\_pass = 1$

No Deadlock (1)

$$
\begin{array}{l}
\quad 0 < d \\
\quad ml\_tl \;\in \{\textsf{red}, \textsf{green}\} \\
\quad il\_tl \;\in \{\textsf{red}, \textsf{green}\} \\
\quad ml\_pass \;\in \{0, 1\} \\
\quad il\_pass \;\in \{0, 1\} \\
\quad a \;\in\; \mathbb{N} \\
\quad b \;\in\; \mathbb{N} \\
\quad c \;\in\; \mathbb{N} \\
\quad ml\_tl = \textsf{red} \;\Rightarrow\; ml\_pass = 1 \\
\quad il\_tl = \textsf{red} \;\Rightarrow\; il\_pass = 1 \\
\Rightarrow \\
\quad (ml\_tl = \textsf{red} \;\wedge\; a + b < d \;\wedge\; c = 0 \;\wedge\; il\_pass = 1) \;\vee \\
\quad (il\_tl = \textsf{red} \;\wedge\; a = 0 \;\wedge\; b > 0 \;\wedge\; ml\_pass = 1) \;\vee \\
\quad ml\_tl = \textsf{green} \;\vee\; il\_tl = \textsf{green} \;\vee\; a > 0 \;\vee\; c > 0
\end{array}
$$

No Deadlock (2)

The previous statement reduces to the following, which is true

$$
\begin{array}{l}
\quad 0 < d \\
\quad a \in \mathbb{N} \\
\quad b \in \mathbb{N} \\
\quad c \in \mathbb{N} \\
\Rightarrow \\
\quad (a + b < d \;\wedge\; c = 0) \;\vee \\
\quad (a = 0 \;\wedge\; b > 0) \;\vee \\
\quad a > 0 \;\vee \\
\quad c > 0
\end{array}
\quad \rightsquigarrow \quad
\begin{array}{l}
\quad 0 < d \\
\quad b \in \mathbb{N} \\
\Rightarrow \\
\quad b < d \;\vee\; b > 0
\end{array}
$$

Second Refinement: Conclusion

- 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

Axioms Abstract invariants Concrete invariants Concrete guards $\vdash$ Same actions on common variables	SIM

Summary of Second Refinement: the State (1)

variables: $a, b, c,$
$ml\_tl, il\_tl, ml\_pass, il\_pass$

inv2_1: $ml\_tl \in \{\text{red}, \text{green}\}$

inv2_2: $il\_tl \in \{\text{red}, \text{green}\}$

inv2_3: $ml\_tl = 1 \Rightarrow a + b < d \wedge c = 0$

inv2_4: $il\_tl = 1 \Rightarrow 0 < b \wedge a = 0$

Summary of Second Refinement: the State (2)

inv2_5: $ml\_tl = \textbf{red} \;\lor\; il\_tl = \textbf{red}$

inv2_6: $ml\_pass \in \{0, 1\}$

inv2_7: $il\_pass \in \{0, 1\}$

inv2_8: $ml\_tl = \textbf{red} \;\Rightarrow\; ml\_pass = 1$

inv2_9: $il\_tl = \textbf{red} \;\Rightarrow\; il\_pass = 1$

variant2: $ml\_pass + il\_pass$

Summary of Second Refinement: the Event (1)

ML_out_1
when
$ml\_tl = \mathbf{green}$
$a + 1 + b < d$
then
$a := a + 1$
$ml\_pass := 1$
end

ML_out_2
when
$ml\_tl = \mathbf{green}$
$a + 1 + b = d$
then
$a := a + 1$
$ml\_pass := 1$
$ml\_tl := \mathbf{red}$
end

Summary of Second Refinement: the Event (2)

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

Summary of Second Refinement: the Event (3)

ML_tl_green
  when
    $ml\_tl = \textbf{red}$
    $a + b < d$
    $c = 0$
    $il\_pass = 1$
  then
    $ml\_tl := \textbf{green}$
    $il\_tl := \textbf{red}$
    $ml\_pass := 0$
  end

IL_tl_green
  when
    $il\_tl = \textbf{red}$
    $0 < b$
    $a = 0$
    $ml\_pass = 1$
  then
    $il\_tl := \textbf{green}$
    $ml\_tl := \textbf{red}$
    $il\_pass := 0$
  end

Summary of Second Refinement: the Event (4)

- These events are identical to their abstract versions

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

IL_in
  when
    $0 < a$
  then
    $a := a - 1$
    $b := b + 1$
  end

Our Refinement Strategy

- Initial model: Limiting the number of cars (FUN_2)

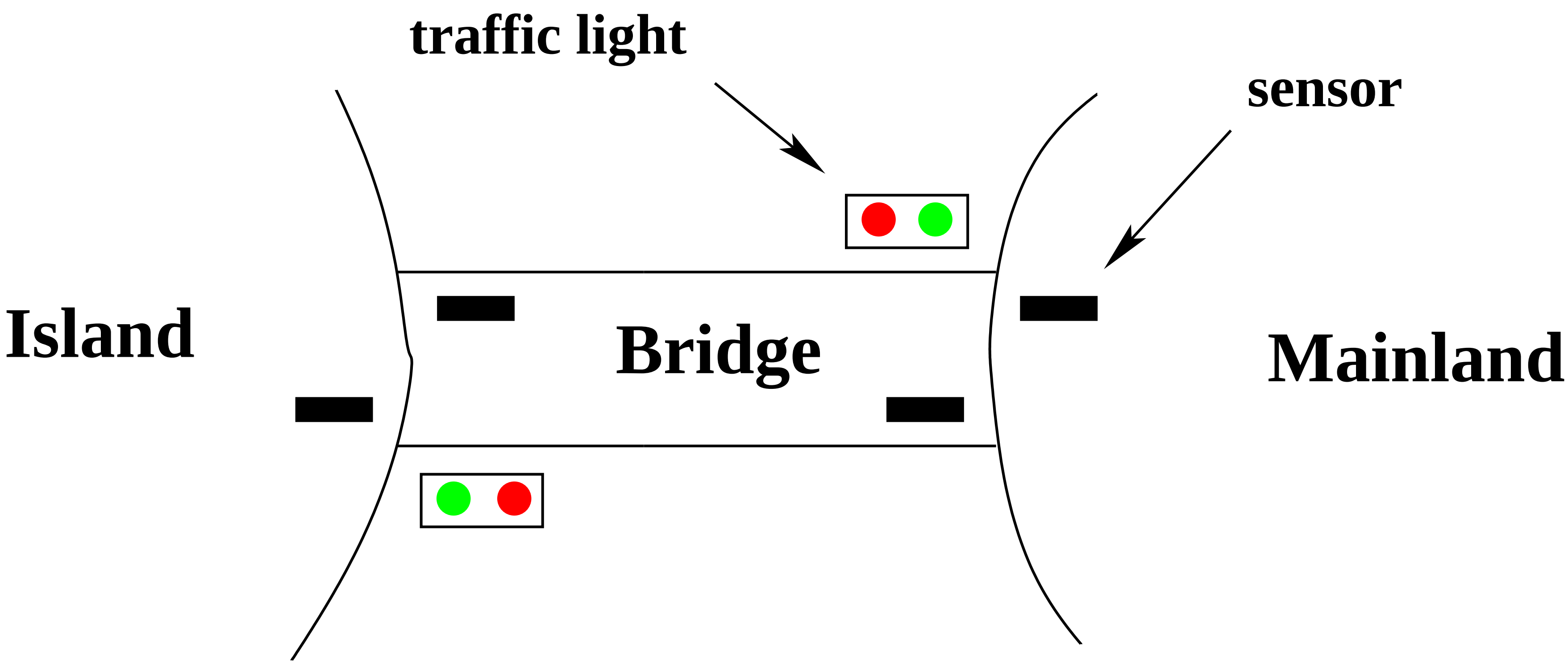
- First refinement: Introducing the one way bridge (FUN_3)

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

- Third refinement: Introducing the sensors (EQP_4,5)

Third Refinement: Adding Car Sensors

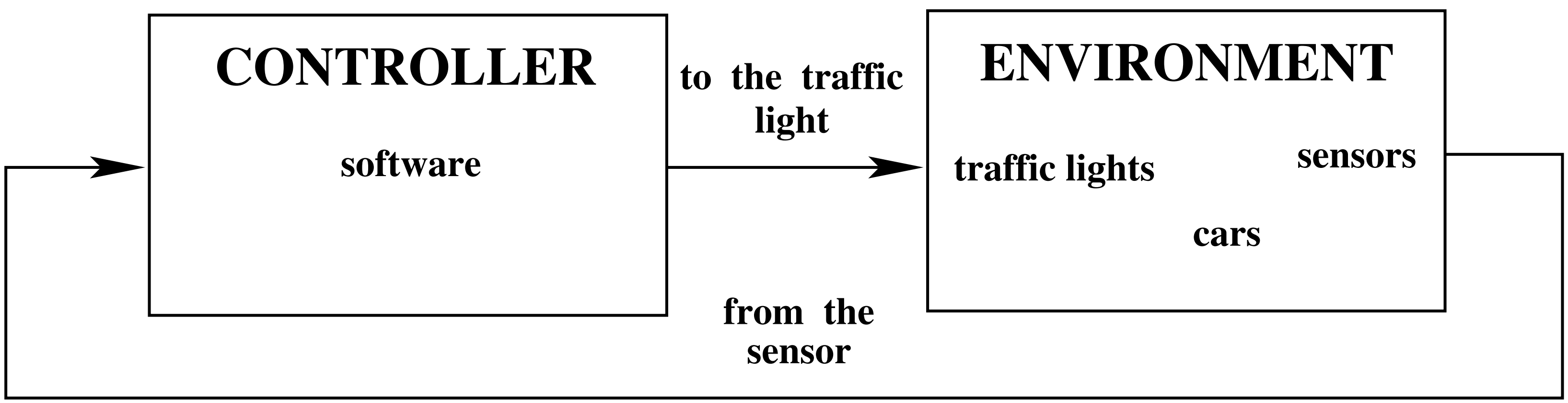
Reminder of the physical system



Closed Model

-We want to clearly identify in our model:

- The controller
- The environment
- The communication channels between the two



Controller Variables

Contoller variables: $\boldsymbol{a}$,

$\boldsymbol{b}$,

$\boldsymbol{c}$,

$\boldsymbol{ml\_pass}$,

$\boldsymbol{il\_pass}$

Environment Variables

These new variables denote physical objects

Environment variables: $\boldsymbol{A}$,

$\boldsymbol{B}$,

$\boldsymbol{C}$,

$\boldsymbol{ML\_OUT\_SR}$,

$\boldsymbol{ML\_IN\_SR}$,

$\boldsymbol{IL\_OUT\_SR}$,

$\boldsymbol{IL\_IN\_SR}$

Output Channel Variables

Output channels: $\boldsymbol{ml\_tl}$,

$\boldsymbol{il\_tl}$

Output Channel Variables

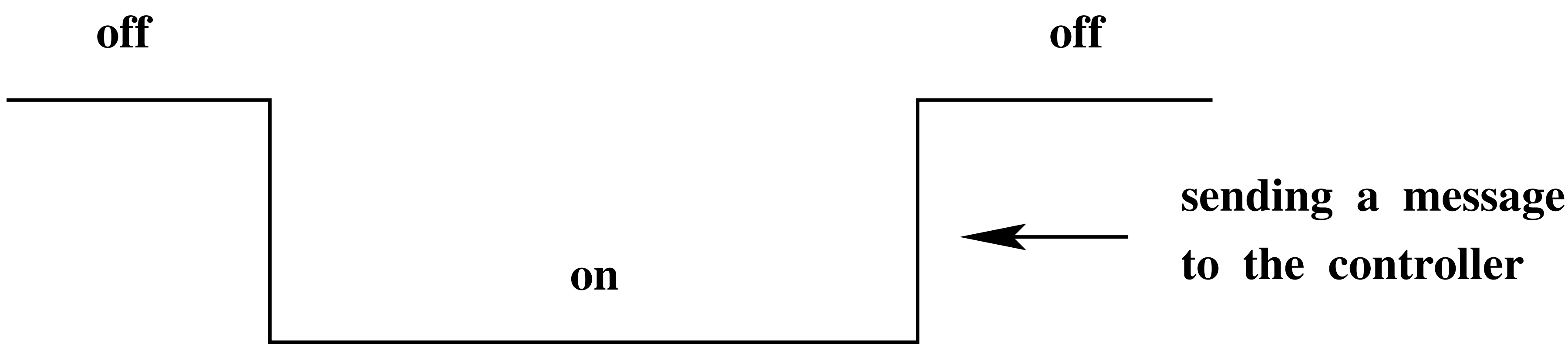
Input channels: ml_out_10,

ml_in_10,

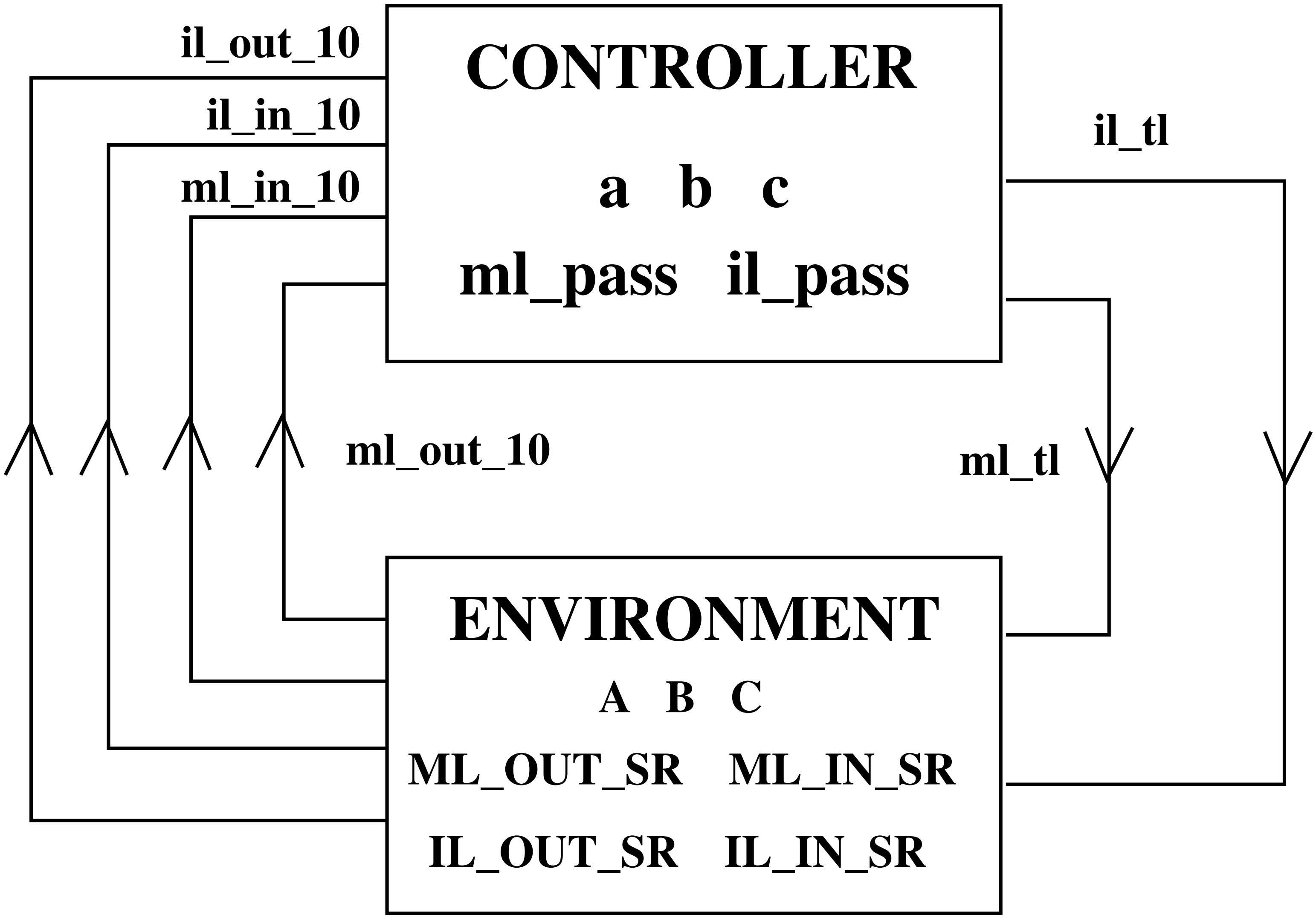
il_in_10,

il_out_10

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



Summary



il_out_10
il_in_10
ml_in_10
CONTROLLER
a b c
ml_pass il_pass
il_tl
ml_out_10
ml_tl
ENVIRONMENT
A B C
ML_OUT_SR ML_IN_SR
IL_OUT_SR IL_IN_SR

Constants

carrier sets: $\ldots, SENSOR$

constants: $\ldots, on, off$

axm3_1: $SENSOR = \{on, off\}$

axm3_2: $on \neq off$

Variables (1)

$$\textbf{inv3\_1}: \quad \boldsymbol{ML\_OUT\_SR} \in \boldsymbol{SENSOR}$$

$$\textbf{inv3\_2}: \quad \boldsymbol{ML\_IN\_SR} \in \boldsymbol{SENSOR}$$

$$\textbf{inv3\_3}: \quad \boldsymbol{IL\_OUT\_SR} \in \boldsymbol{SENSOR}$$

$$\textbf{inv3\_4}: \quad \boldsymbol{IL\_IN\_SR} \in \boldsymbol{SENSOR}$$

Variables (2)

$$\mathbf{inv3\_5:}\quad \boldsymbol{A} \in \mathbb{N}$$

$$\mathbf{inv3\_6:}\quad \boldsymbol{B} \in \mathbb{N}$$

$$\mathbf{inv3\_7:}\quad \boldsymbol{C} \in \mathbb{N}$$

$$\mathbf{inv3\_8:}\quad \boldsymbol{ml\_out\_10} \in \mathbf{BOOL}$$

$$\mathbf{inv3\_9:}\quad \boldsymbol{ml\_in\_10} \in \mathbf{BOOL}$$

$$\mathbf{inv3\_10:}\quad \boldsymbol{il\_out\_10} \in \mathbf{BOOL}$$

$$\mathbf{inv3\_11:}\quad \boldsymbol{il\_in\_10} \in \mathbf{BOOL}$$

Invariants (1)

When sensors are on, there are cars on them

$$\mathbf{inv3\_12}: \quad IL\_IN\_SR = on \;\Rightarrow\; A > 0$$

$$\mathbf{inv3\_13}: \quad IL\_OUT\_SR = on \;\Rightarrow\; B > 0$$

$$\mathbf{inv3\_14}: \quad ML\_IN\_SR = on \;\Rightarrow\; C > 0$$

The sensors are used to detect the presence of cars entering or leaving the bridge	EQP-5

Invariants (2)

Drivers obey the traffic lights

$$\mathbf{inv3\_15:} \quad \boldsymbol{ml\_out\_10} = \mathbf{TRUE} \;\Rightarrow\; \boldsymbol{ml\_tl} = \boldsymbol{green}$$

$$\mathbf{inv3\_16:} \quad \boldsymbol{il\_out\_10} = \mathbf{TRUE} \;\Rightarrow\; \boldsymbol{il\_tl} = \boldsymbol{green}$$

Cars are not supposed to pass on a red traffic light, only on a green one	EQP-3

Invariants (3)

When a sensor is "on", the previous information is treated

$$\mathbf{inv3\_17:} \quad \boldsymbol{IL\_IN\_SR = on \;\Rightarrow\; il\_in\_10 = \mathrm{FALSE}}$$

$$\mathbf{inv3\_18:} \quad \boldsymbol{IL\_OUT\_SR = on \;\Rightarrow\; il\_out\_10 = \mathrm{FALSE}}$$

$$\mathbf{inv3\_19:} \quad \boldsymbol{ML\_IN\_SR = on \;\Rightarrow\; ml\_in\_10 = \mathrm{FALSE}}$$

$$\mathbf{inv3\_20:} \quad \boldsymbol{ML\_OUT\_SR = on \;\Rightarrow\; ml\_out\_10 = \mathrm{FALSE}}$$

The controller must be fast enough so as to be able to treat all the information coming from the environment	FUN-5

Invariants (4)

Linking the physical and logical cars (1)

$$\mathbf{inv3\_21:}\quad il\_in\_10 = \mathrm{TRUE} \;\wedge\; ml\_out\_10 = \mathrm{TRUE} \;\Rightarrow\; A = a$$

$$\mathbf{inv3\_22:}\quad il\_in\_10 = \mathrm{FALSE} \;\wedge\; ml\_out\_10 = \mathrm{TRUE} \;\Rightarrow\; A = a + 1$$

$$\mathbf{inv3\_23:}\quad il\_in\_10 = \mathrm{TRUE} \;\wedge\; ml\_out\_10 = \mathrm{FALSE} \;\Rightarrow\; A = a - 1$$

$$\mathbf{inv3\_24:}\quad il\_in\_10 = \mathrm{FALSE} \;\wedge\; ml\_out\_10 = \mathrm{FALSE} \;\Rightarrow\; A = a$$

Linking the physical and logical cars (2)

$$\begin{array}{ll}
\textbf{inv3_25}: & il_in_10 = \mathrm{TRUE} \;\wedge\; il_out_10 = \mathrm{TRUE} \;\;\Rightarrow\;\; B = b \\
\textbf{inv3_26}: & il_in_10 = \mathrm{TRUE} \;\wedge\; il_out_10 = \mathrm{FALSE} \;\;\Rightarrow\;\; B = b + 1 \\
\textbf{inv3_27}: & il_in_10 = \mathrm{FALSE} \;\wedge\; il_out_10 = \mathrm{TRUE} \;\;\Rightarrow\;\; B = b - 1 \\
\textbf{inv3_28}: & il_in_10 = \mathrm{FALSE} \;\wedge\; il_out_10 = \mathrm{FALSE} \;\;\Rightarrow\;\; B = b
\end{array}$$

$$\begin{array}{ll}
\textbf{inv3_29}: & il_out_10 = \mathrm{TRUE} \;\wedge\; ml_out_10 = \mathrm{TRUE} \;\;\Rightarrow\;\; C = c \\
\textbf{inv3_30}: & il_out_10 = \mathrm{TRUE} \;\wedge\; ml_out_10 = \mathrm{FALSE} \;\;\Rightarrow\;\; C = c + 1 \\
\textbf{inv3_31}: & il_out_10 = \mathrm{FALSE} \;\wedge\; ml_out_10 = \mathrm{TRUE} \;\;\Rightarrow\;\; C = c - 1 \\
\textbf{inv3_32}: & il_out_10 = \mathrm{FALSE} \;\wedge\; ml_out_10 = \mathrm{FALSE} \;\;\Rightarrow\;\; C = c
\end{array}$$

Invariants (6)

The basic properties hold for the physical cars

$$\textbf{inv3\_33}: \quad A = 0 \quad \lor \quad C = 0$$

$$\textbf{inv3\_34}: \quad A + B + C \leq d$$

The number of cars on the bridge and the island is limited	FUN-2

The bridge is one way or the other, not both at the same time	FUN-3

Refining Abstract Events (1)

ML_out_1
when
$\underline{ml\_out\_10 = \text{TRUE}}$
$a + b + 1 \neq d$
then
$a := a + 1$
$ml\_pass := 1$
$\underline{ml\_out\_10 := \text{FALSE}}$
end

ML_out_2
when
$\underline{ml\_out\_10 = \text{TRUE}}$
$a + b + 1 = d$
then
$a := a + 1$
$ml\_tl := red$
$ml\_pass := 1$
$\underline{ml\_out\_10 := \text{FALSE}}$
end

(abstract-)ML_out_1
when
$ml\_tl = \textbf{green}$
$a + b + 1 \neq d$
then
$a := a + 1$
$ml\_pass := 1$
end

(abstract-)ML_out_2
when
$ml\_tl = \textbf{green}$
$a + b + 1 = d$
then
$a := a + 1$
$ml\_pass := 1$
$ml\_tl := \textbf{red}$
end

Refining Abstract Events (2)

IL_out_1
when
$\underline{il\_out\_10 = \mathrm{TRUE}}$
$b \neq 1$
then
$b := b - 1$
$c := c + 1$
$il\_pass := 1$
$\underline{il\_out\_10 := \mathrm{FALSE}}$
end

IL_out_2
when
$\underline{il\_out\_10 = \mathrm{TRUE}}$
$b = 1$
then
$b := b - 1$
$c := c + 1$
$il\_tl := red$
$il\_pass := 1$
$\underline{il\_out\_10 := \mathrm{FALSE}}$
end

(abstract-)IL_out_1
when
$il_tl = $ **green**
$b \neq 1$
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)


```
ML_in
  when
    ml_in_10 = TRUE
    0 < c
  then
    c := c − 1
    ml_in_10 := FALSE
  end
```

```
IL_in
  when
    il_in_10 = TRUE
    0 < a
  then
    a := a − 1
    b := b + 1
    il_in_10 := FALSE
  end
```

```
(abstract-)ML_in
  when
    0 < c
  then
    c := c − 1
  end
```

```
(abstract-)IL_in
  when
    0 < a
  then
    a := a − 1
    b := b + 1
  end
```

Refining Abstract Events (4)

ML_tl_green
when
$ml\_tl = red$
$a + b < d$
$c = 0$
$il\_pass = 1$
$\underline{il\_out\_10 = \text{FALSE}}$
then
$ml\_tl := green$
$il\_tl := red$
$ml\_pass := \text{FALSE}$
end

IL_tl_green
when
$il\_tl = red$
$a = 0$
$ml\_pass = 1$
$\underline{ml\_out\_10 = \text{FALSE}}$
then
$il\_tl := green$
$ml\_tl := red$
$il\_pass := \text{FALSE}$
end

(abstract-)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

(abstract-)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

Adding New PHYSICAL Events (1)

ML_out_arr
when
$ML\_OUT\_SR = off$
$ml\_out\_10 = \mathrm{FALSE}$
then
$ML\_OUT\_SR := on$
end

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

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

IL_out_arr
when
$IL\_OUT\_SR = off$
$il\_out\_10 = \mathrm{FALSE}$
$B > 0$
then
$IL\_OUT\_SR := on$
end

Adding New PHYSICAL Events (2)

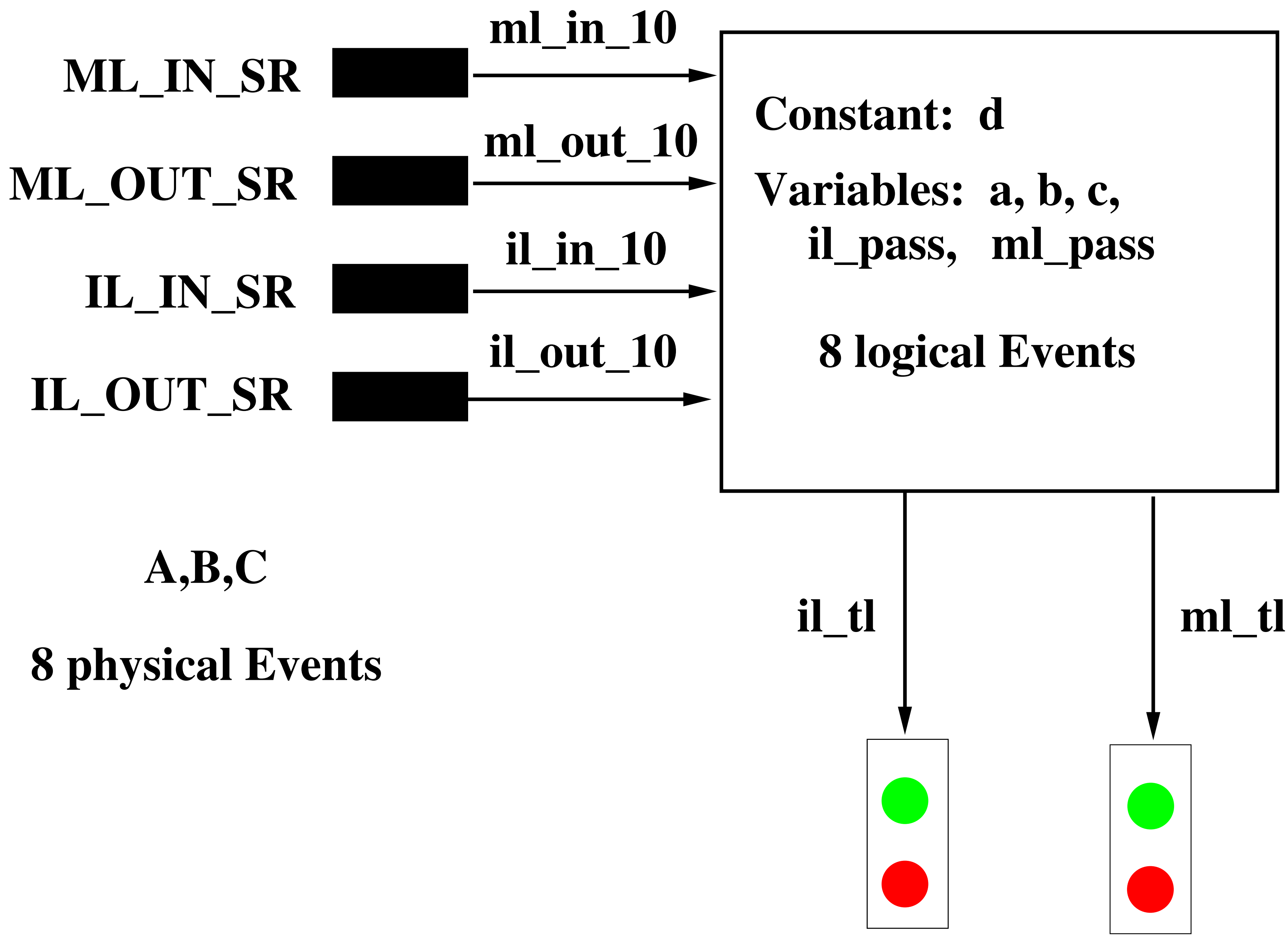
ML_out_dep
when
$ML\_OUT\_SR = on$
$ml\_tl = green$
then
$ML\_OUT\_SR := off$
$ml\_out\_10 := \mathrm{TRUE}$
$A := A + 1$
end

ML_in_dep
when
$ML\_IN\_SR = on$
then
$ML\_IN\_SR := off$
$ml\_in\_10 := \mathrm{TRUE}$
$C := C - 1$
end

IL_in_dep
when
$IL\_IN\_SR = on$
then
$IL\_IN\_SR := off$
$il\_in\_10 := \mathrm{TRUE}$
$A := A - 1$
$B := B + 1$
end

IL_out_dep
when
$IL\_OUT\_SR = on$
$il\_tl = green$
then
$IL\_OUT\_SR := off$
$il\_out\_10 := \mathrm{TRUE}$
$B := B - 1$
$C := C + 1$
end

Final Structure of the Controller



Questions on Proving

- 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)

About Tools

- 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 252 proofs
  - Initial model: 6 (0)
  - 1st refinement: 27 (0)
  - 2nd refinement: 81 (1)
  - 3rd refinement: 138 (3)

- All proved automatically (except 4) by the Rodin Platform

Summary of Mathematical Notations (1)

$P \wedge Q$	conjunction
$P \vee Q$	disjunction
$P \Rightarrow Q$	implication
$\neg P$	negation
$x \in S$	set membership operator

Summary of Mathematical Notations (2)

$\mathbb{N}$	set of Natural Numbers: $\{0, 1, 2, 3, \ldots\}$
$\mathbb{Z}$	set of Integers: $\{0, 1, -1, 2, -2, \ldots\}$
$\{a, b, \ldots\}$	set defined in extension
$a + b$	addition of a and b
$a - b$	subtraction of a and b

Summary of Mathematical Notations (3)

$a * b$	product of a and b
$a = b$	equality relation
$a \leq b$	smaller than or equal relation
$a < b$	smaller than relation

Invariant Establishment Proof Rule

- For the init event in the initial model

Axioms of the constants $\Rightarrow$ Modified Invariants	INV

Invariant Preservation Proof Rule

- For other events in the initial model

Axioms of the constants Invariants Guard of the event $\Rightarrow$ Modified Invariants	INV

Deadlock Freeness Rule

- This rule is not mandatory

Axiom of the constant Invariants $\Rightarrow$ Disjunction of the guards	DLF

Refinement Rules (1): Guard Strengthening

- For old events only

Axioms of the constants Abstract invariants Concrete invariants Concrete guards $\Rightarrow$ Abstract guards	GRD

Refinement Rules (2): Invariant Establishment

- For init event only

Axioms of the constants $\Rightarrow$ Modified concrete invariants	INV

Refinement Rules (3): Invariant Preservation

- For all events (except init)

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

Axioms of the constants Abstract invariant Concrete invariant Concrete guard $\Rightarrow$ Modified concrete invariant	INV

Refinement Rules (4): Non-divergence of New Events

- For new events only

Axioms of the constants Abstract invariants Concrete invariants Concrete guard of a new event $\Rightarrow$ Variant $\in \mathbb{N}$	NAT

Refinement Rules (5): Non-divergence of New Events

- For new events only

Axioms
Abstract invariants
Concrete invariants
Concrete guard
$\vdash$
Modified Var. $<$ Var.

Refinement Rules (6): Relative Deadlock Freeness

- Global proof rule

Axioms of the constants Abstract invariants Concrete invariants Disjunction of abstract guards $\Rightarrow$ Disjunction of concrete guards	DLF

Refinement Rules (7)

- For old events (in case of superposition)

Axioms of constants Abstract invariants Concrete invariants Concrete guards $\Rightarrow$ Same actions on common variables	SIM