

2. Controlling Cars on a Bridge

Jean-Raymond Abrial

April 2008

- To present an **example of system development**
- Our approach: a series of **more and more accurate models**
- This approach is called **refinement**
- The models formalize the view of an **external observer**

- Each model will be analyzed and **proved to be correct**
- The **aim** is to obtain a system that will be **correct by construction**
- The **correctness criteria** correspond to a number of **proof rules**
- **Proofs** will be performed by using the **sequent calculus**
- **Inference rules** used in the sequent calculus will be **reviewed**

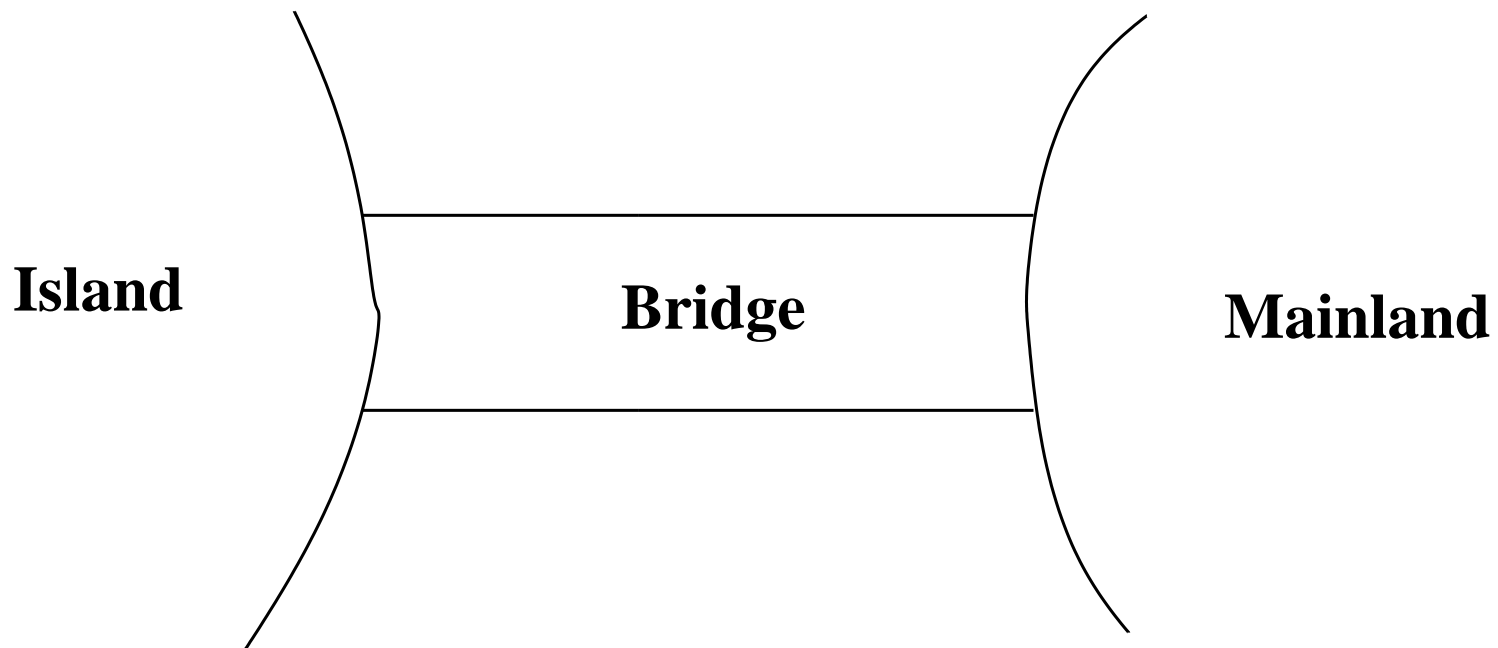
- The concepts of **state** and **events** for defining models
- A few **principles** of system development: essentially **refinement**
- A refresher on **classical logic** and **simple arithmetic foundations**
- A refresher on **formal proofs**

1. Presenting the **requirement document** (as in previous lecture)
 2. Defining the **strategy**
 3. Development of the **initial model** and the **refinements**
- Remark: During the development some **theoretical background** will be provided

- The system we are going to build is a piece of software connected to some equipment.
- There are two kinds of requirements:
 - those concerned with the equipment, labeled EQP,
 - those concerned with the function of the system, labeled FUN.
- The function of this system is to control cars on a narrow bridge.
- This bridge is supposed to link the mainland to a small island.

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

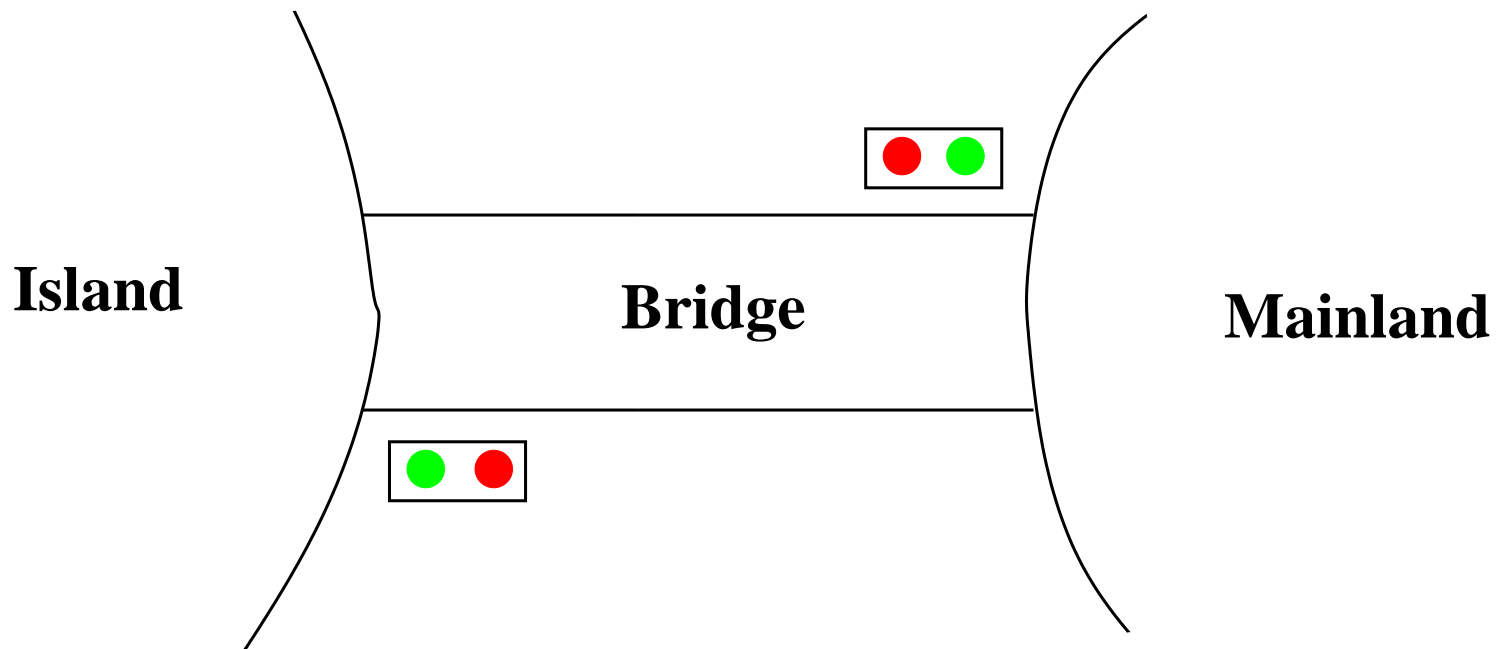
- This can be illustrated as follows



- The controller is equipped with two traffic lights with two colors.

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

- One of the traffic lights is situated on the mainland and the other one on the island. Both are close to the bridge.
- This can be illustrated as follows



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

EQP-2

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

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

EQP-3

- There are also some car sensors situated at both ends of the bridge.
- These sensors are supposed to detect the presence of cars intending to enter or leave the bridge.
- There are four such sensors. Two of them are situated on the bridge and the other two are situated on the mainland and on the island.

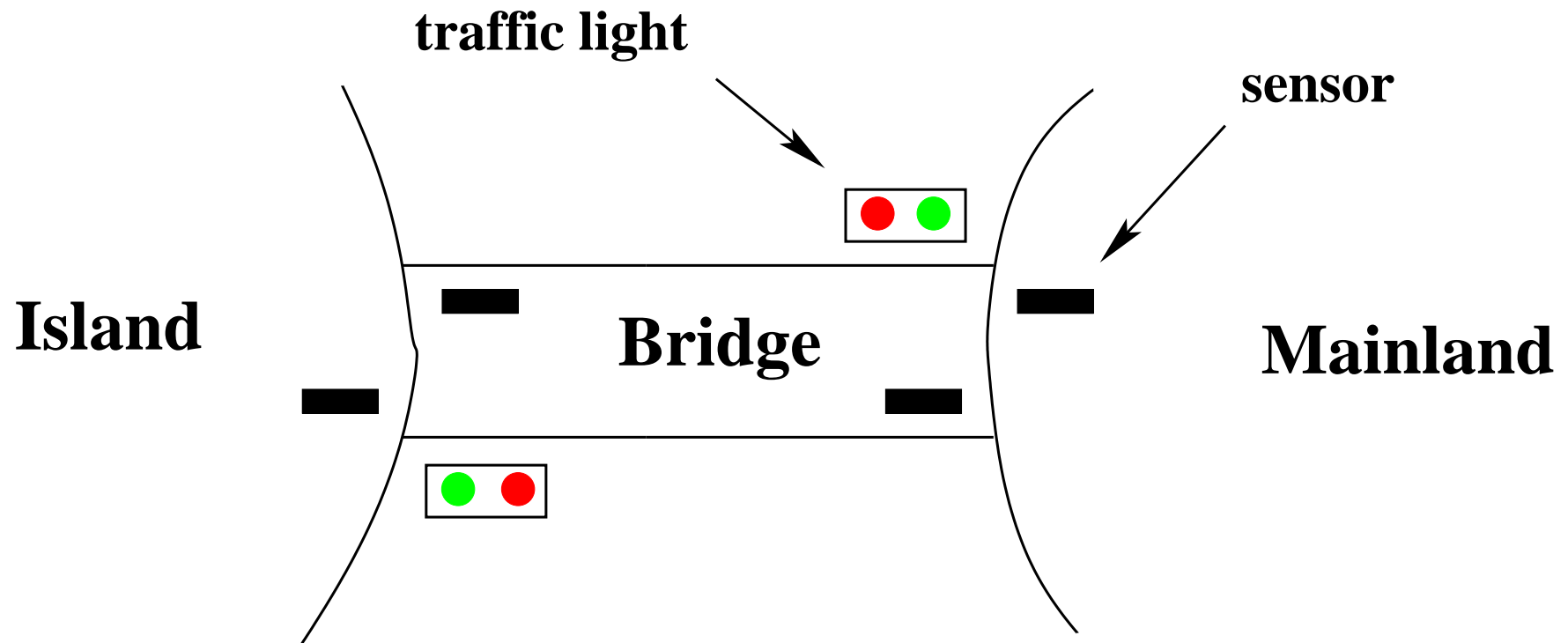
The system is equipped with four car sensors each with two states: on or off

EQP-4

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

EQP-5

- The pieces of equipment can be illustrated as follows:



- This system has two main constraints: the number of cars on the bridge and the island is limited and the bridge is one way.

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

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

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

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

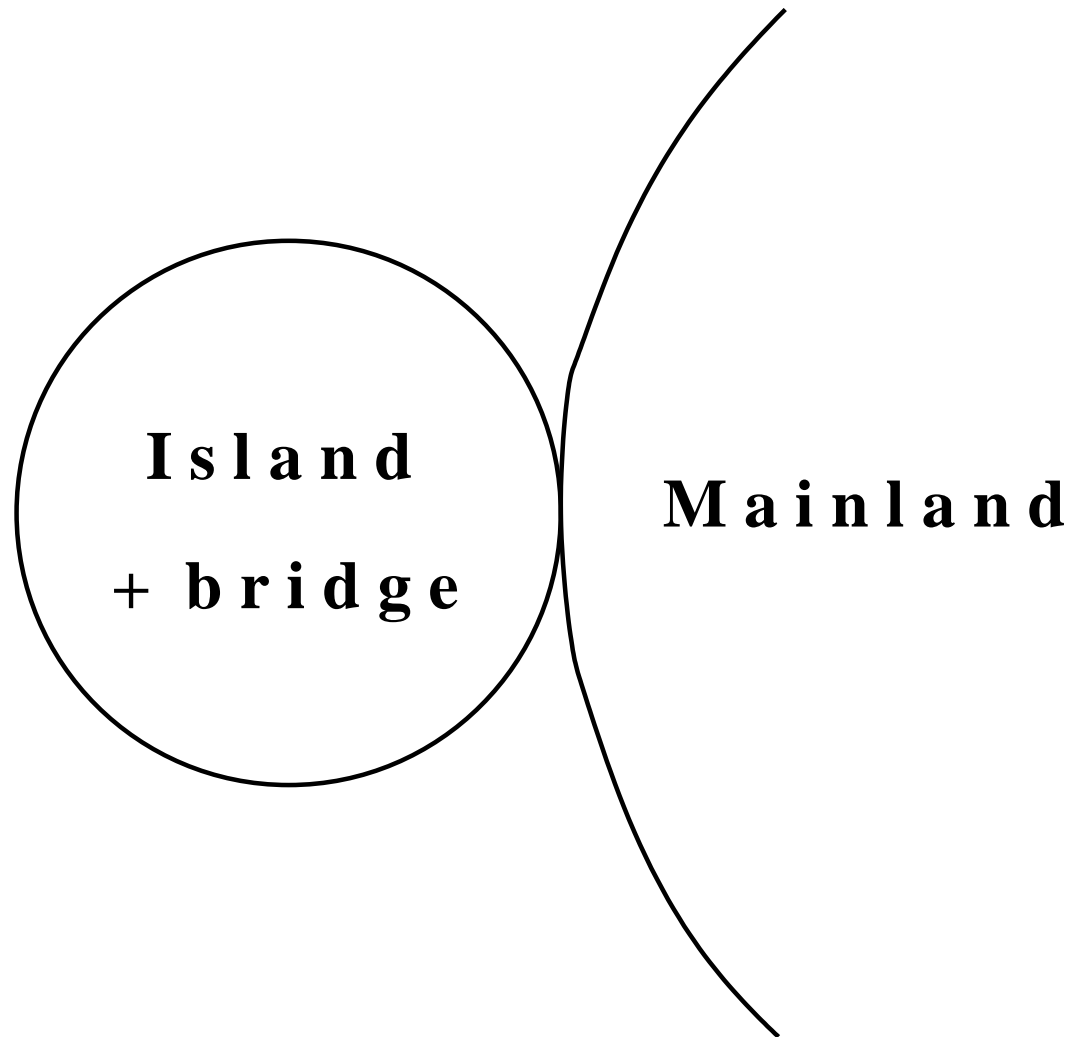
EQP-4

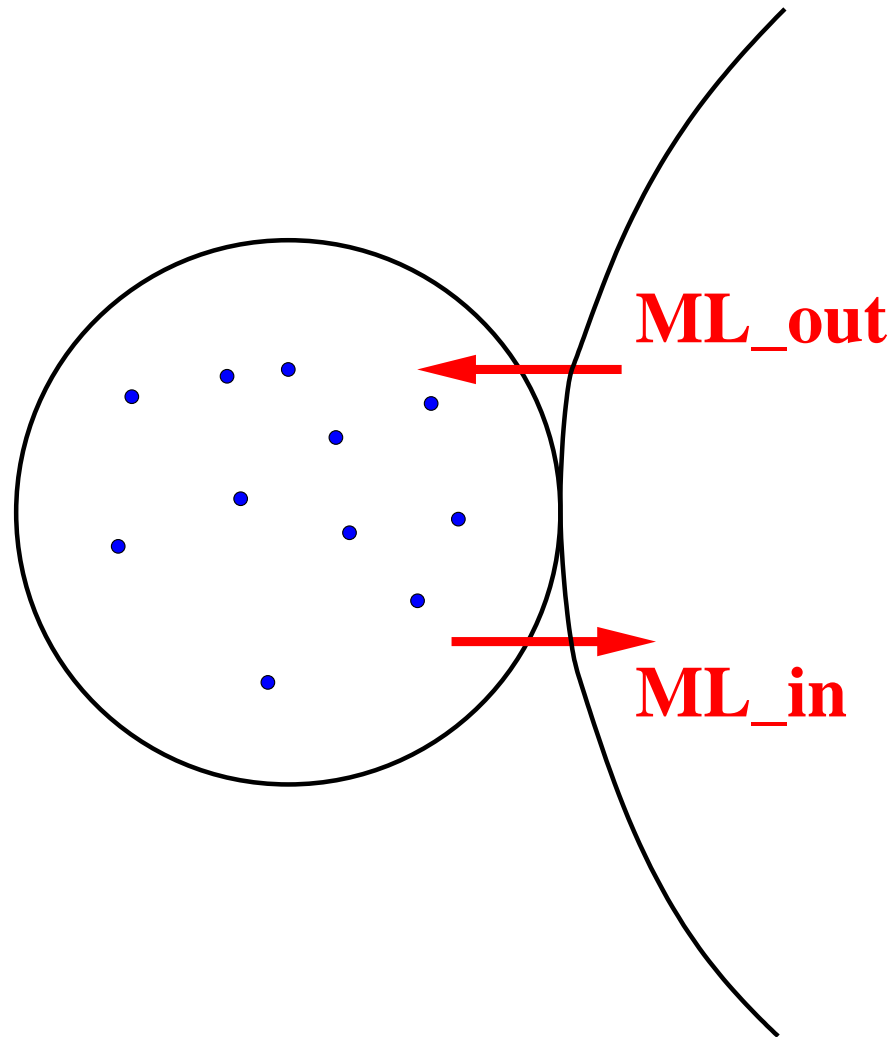
The sensors are used to detect the presence of cars entering or leaving the bridge

EQP-5

- **Initial model**: Limiting the number of cars (FUN_2)
- **First refinement**: Introducing the one way bridge (FUN_3)
- **Second refinement**: Introducing the traffic lights (EQP_1,2,3)
- **Third refinement**: Introducing the sensors (EQP_4,5)

- It is **very simple**
- We do not consider at all the equipment: traffic lights and sensors
- We even do not consider the bridge
- We are just interested in the **pair “island-bridge”**
- We are focusing on **FUN-2** (limited number of cars on island-bridge)





- **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 in the Island-Bridge
- **axm0_1** states that d is a **natural number**
- Constant d is a member of the set $\mathbb{N} = \{0, 1, 2, , \dots\}$

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

variable: n

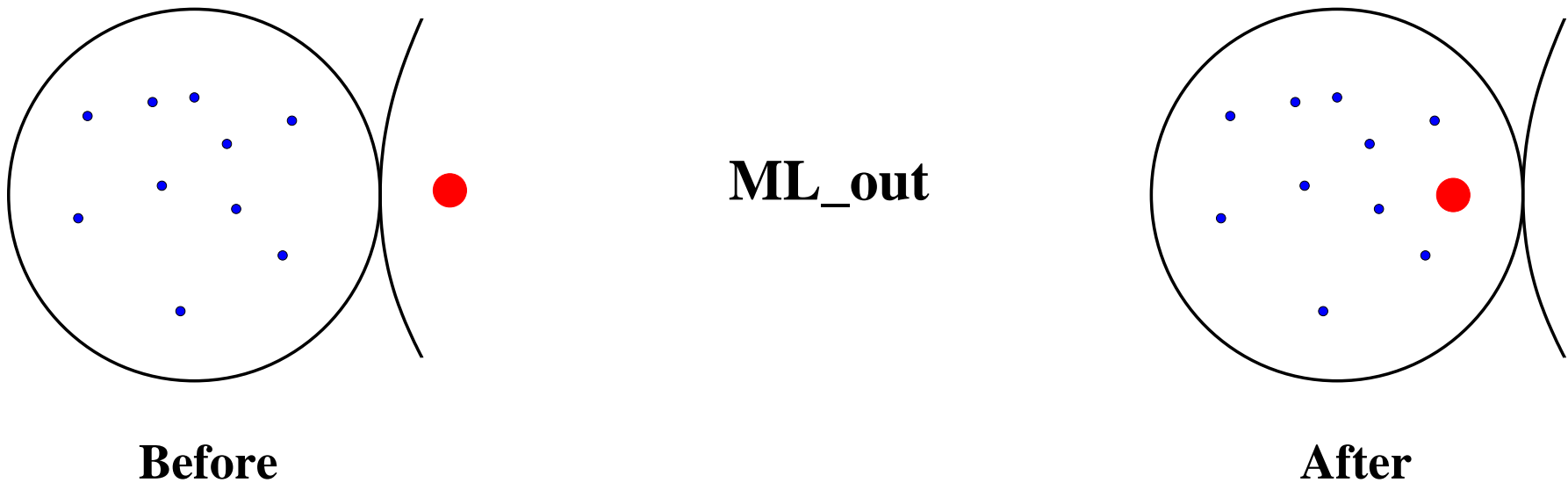
inv0_1: $n \in \mathbb{N}$

inv0_2: $n \leq d$

- n is the **effective number of cars** in 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**

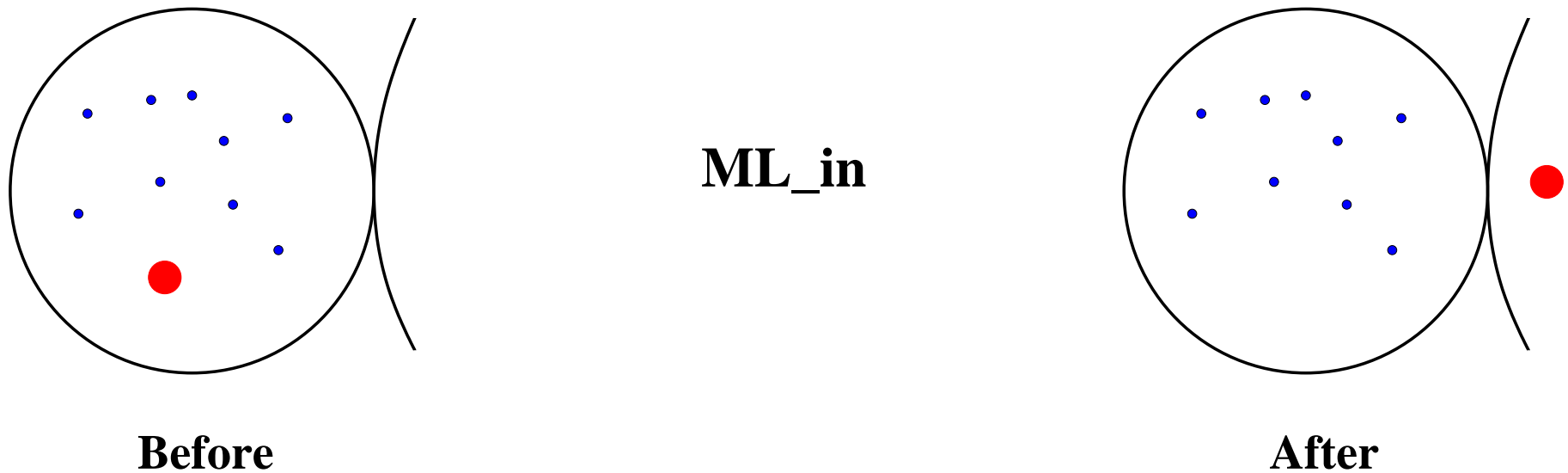
- Labels **axm0_1**, **inv0_1**, etc. have been chosen **systematically**
- **axm** or **inv** recall the **purpose**: **axioms** of constants or **invariants** of variables
- The **0** as in **inv0_1** stands for the initial model.
- Later we shall have **inv1_1** for an invariant of refinement **1**, etc.
- The **1** as in **inv0_1** is a serial number
- Any convention is **valid** as long as it is **systematic**

- This is the **first transition** (or event) that can be **observed**
- A car is leaving the mainland and entering the Island-Bridge



- The **number of cars** in the Island-Bridge has been **incremented**

- We can also observe a **second transition** (or event)
- A car leaving the Island-Bridge and re-entering the mainland



- The **number of cars** in the Island-Bridge has been **decremented**

- Event 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)

- These events are approximations for **two reasons**:
 1. They might be **refined** (made more precise) later
 2. They might be **insufficient** at this stage because **not consistent with the invariant**
- We shall have to perform a **proof** in order to **check this consistency**

- To each event can be associated a **before-after predicate**
- It denotes the **relationship** holding between the **values** of the variable *just before* and *just after* the event occurrence
- The **before-value** is denoted by the **variable name**, say n
- The **after-value** is denoted by the **variable name primed**, say n'

The Events

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

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

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

- The after-value n' is defined as a **function** of the before-value n
- This is because the corresponding events are **deterministic**
- In further lectures, we shall consider some **non-deterministic** events:

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

- When we wrote events **ML_out** and **ML_in** we did not take into accounts invariants **inv0_1** and **inv0_2**
- There is **no reason** a priori for these invariants to be preserved
- This has to be **proved rigorously**
- We have to **clarify** what we have to prove
- We are going to define some **proof obligations** also called **verification conditions**

- Let us consider invariant **inv0_1**

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

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

$$n' = n + 1$$

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

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

- Under hypothesis $n \in \mathbb{N}$ then $n + 1 \in \mathbb{N}$ is provable

- This can be written as follows

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

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

- This can be **generalized** (next slide)

- We collectively denote our set of **constants** by c
- We denote our set of **axioms** by $A(c)$: $A_1(c), A_2(c), \dots$
- We collectively denote our set of **variables** by v
- We denote our set of **invariants** by $I(c, v)$: $I_1(c, v), I_2(c, v), \dots$

- We are given an **event** with **before-after predicate** $v' = E(c, v)$
- We have to prove the following in order to **preserve invariant** $I_i(c, v)$:

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

- It says: prove $I_i(c, E(c, v))$ under hypotheses $A(c)$ and $I(c, v)$
- We have given the name **INV** to this proof obligation

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

- Just before occurrence of the event represented by $v' = E(c, v)$, $A(c)$ as well as $I(c, v)$ clearly holds. We can then assume them
- Just after the occurrence, invariant $I_i(c, v)$ has become $I_i(c, v')$, that is $I_i(c, E(c, v))$
- This statement $I_i(c, E(c, v))$ must hold since we claimed that $I_i(c, v)$ was an invariant

- The proof obligation

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

can be re-written vertically as follows:

Axioms Invariants \vdash Modified Invariant	$A(c)$ $I(c, v)$ \vdash $I_i(c, E(c, v))$	INV
--	--	-----

- A **sequent** is a formal statement of the following shape

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

- The symbol " \vdash " is called the **turnstile** (it corresponds to verb "**yield**")
- **H** denotes a set of **predicates**: the **hypotheses** (or **assumptions**)
- **G** denotes a predicate: the **goal** (or **conclusion**)
- It can be read: "**Assumptions H yield conclusion G**"

- We have **two events**

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, **four statements have to be proved**

ML_out

$n := n + 1$

$(n' = n + 1)$

Axiom **axm0_1**

Invariant **inv0_1**

Invariant **inv0_2**

⊢

Modified Invariant **inv0_1**

$d \in \mathbb{N}$

$n \in \mathbb{N}$

$n \leq d$

⊢

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

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

ML_out

$n := n + 1$

$(n' = n + 1)$

Axiom **axm0_1**

Invariant **inv0_1**

Invariant **inv0_2**

⊢

Modified Invariant **inv0_2**

$d \in \mathbb{N}$

$n \in \mathbb{N}$

$n \leq d$

⊢

$n + 1 \leq d$

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

ML_in

$n := n - 1$

$(n' = n - 1)$

Axiom **axm0_1**

Invariant **inv0_1**

Invariant **inv0_2**

⊢

Modified Invariant **inv0_1**

$d \in \mathbb{N}$

$n \in \mathbb{N}$

$n \leq d$

⊢

$n - 1 \in \mathbb{N}$

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

ML_in $n := n - 1$ $(n' = n - 1)$ Axiom **axm0_1**Invariant **inv0_1**Invariant **inv0_2**

⊢

Modified Invariant **inv0_2** $d \in \mathbb{N}$ $n \in \mathbb{N}$ $n \leq d$

⊢

 $n - 1 \leq d$

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

ML_out / **inv0_1** / INV

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

ML_out / **inv0_2** / INV

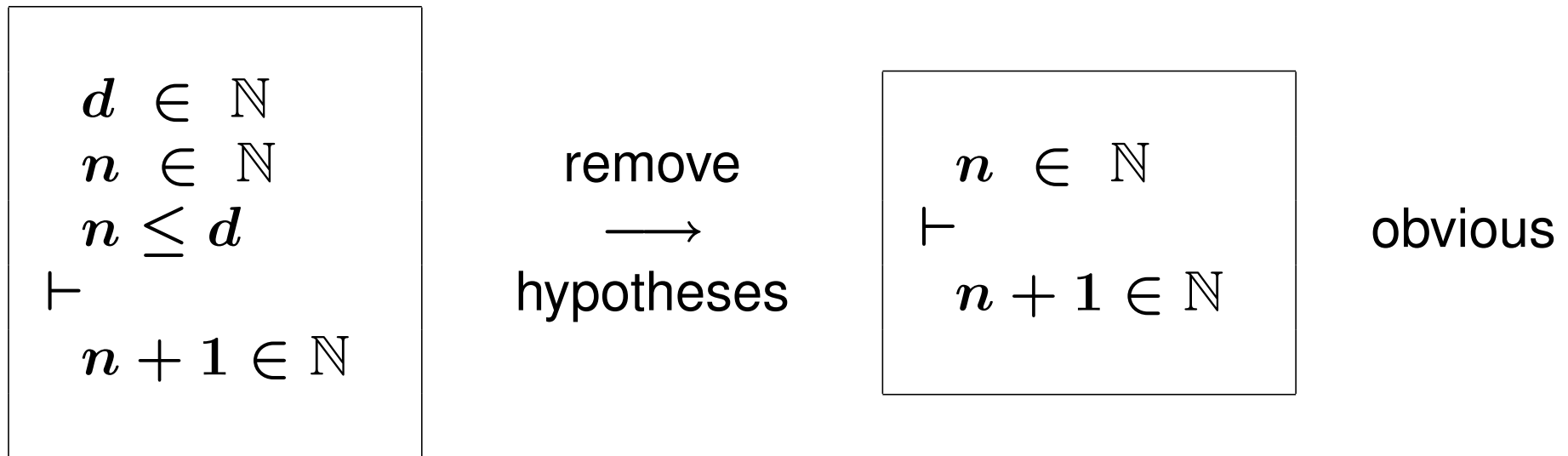
$$\begin{array}{l} d \in \mathbb{N} \\ n \in \mathbb{N} \\ n \leq d \\ \vdash \\ n + 1 \leq d \end{array}$$

ML_in / **inv0_1** / INV

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

ML_in / **inv0_2** / INV

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



- In the first step, we **remove some irrelevant hypotheses**
- In the second and final step, we **accept the sequent as it is**

- In the previous slide, we have applied some **implicit rules**:
 - For **simplifying given sequents**
 - For asserting that a sequent is **proved without justifications**
- In order to be **very precise**, we make these rules **more explicit**
- Such rules are called **rules of inference**

- The first rule of inference can be stated as follows:

$$\frac{\mathbf{H1} \vdash \mathbf{G}}{\mathbf{H1, H2} \vdash \mathbf{G}} \quad \mathbf{MON}$$

- Above the horizontal line, we have a sequent called the **antecedent** (this predicate can be **missing** or on the contrary made **multiple**)
- Below the horizontal line we have a sequent called the **consequent**
- To prove the consequent, **it is sufficient** to prove the antecedent

- The Second **Peano Axiom**

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

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

- Such rules are called **axioms** (because they have **no antecedents**)

- Such rules are given here **without demonstration**

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

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

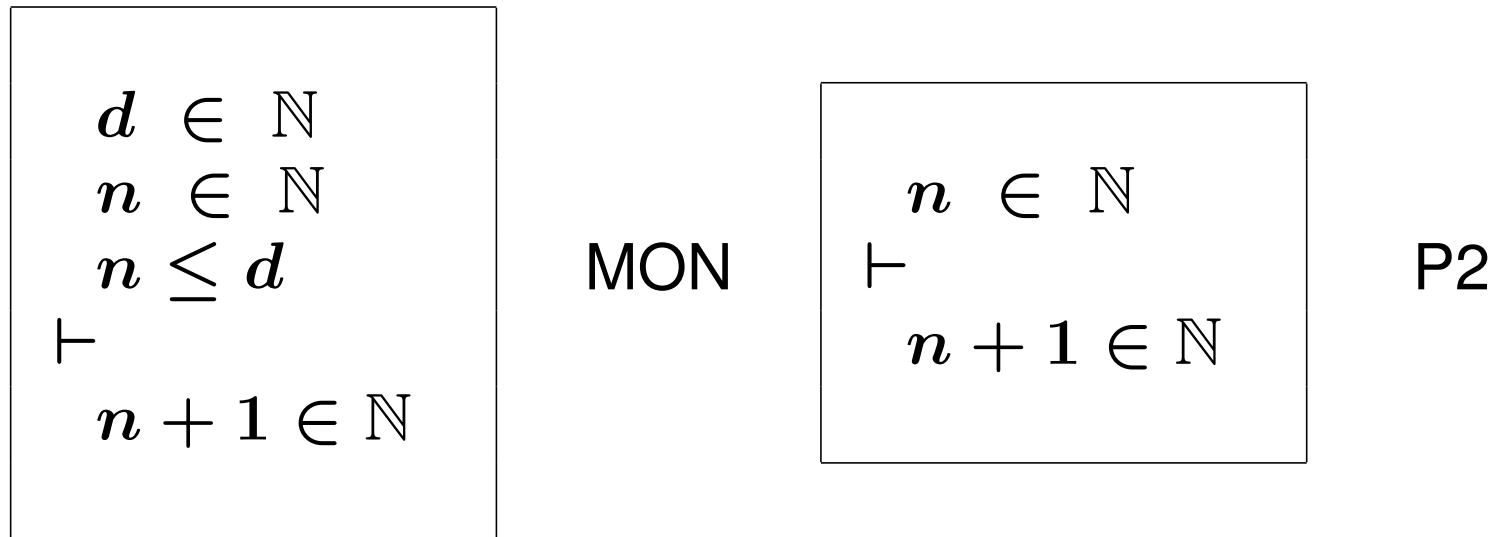
- This rule denotes the **2nd Peano axiom** in very general terms:

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

- It is a **rule schema** where **n** is called a **meta-variable**
- It can be applied to the following sequent by **pattern matching**:

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

- A proof is (for the moment) just a **sequence of sequents**

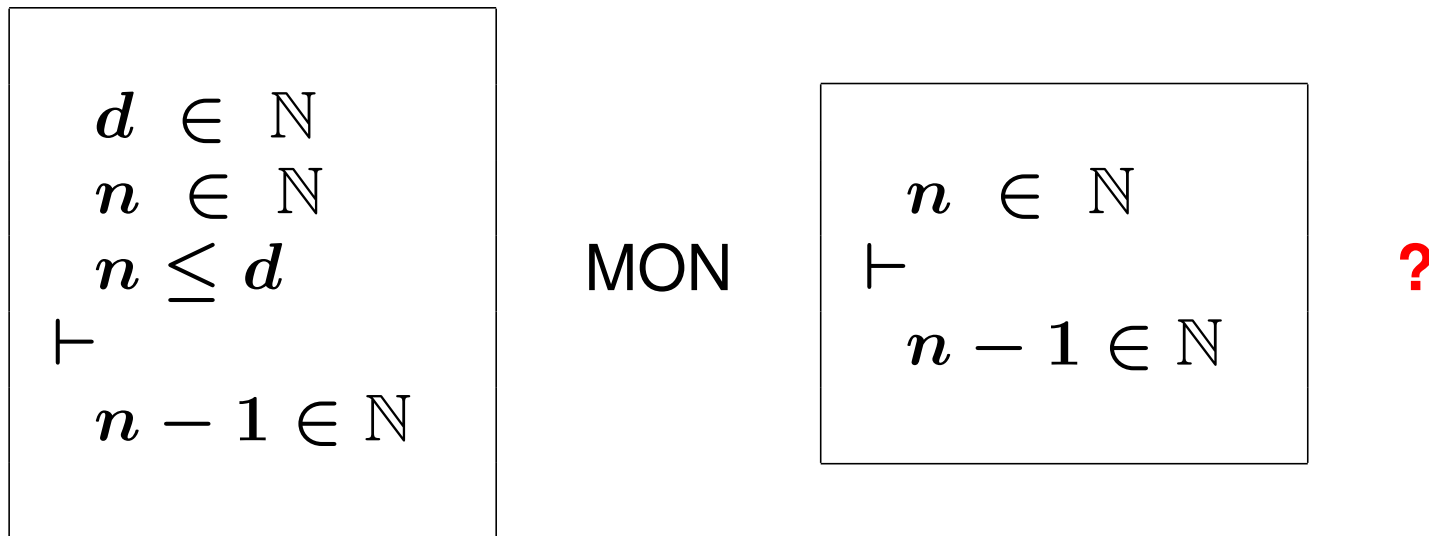


- The sequents are separated by the rule (name) connecting them
- A last rule name (with no antecedent) terminates the sequence

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

- We put a **?** to indicate that **we have no rule to apply**
- The proof fails: we cannot conclude with rule **INC** ($n < d$ needed)

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



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

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

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

MON

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

DEC

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

- 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

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”

- Given c with axioms $A(c)$ and v with invariants $I(c, v)$
- Given an event with guard $G(c, v)$ and b-a predicate $v' = E(c, v)$
- We modify the Invariant Preservation Rule as follows:

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

Axiomss
Invariants
Guard of the event
┆
Modified Invariant

$A(c)$ $I(c, v)$ $G(c, v)$ ┆ $I_i(c, E(c, v))$	INV
--	-----

- Adding a new assumptions to a sequent **does not modify a proof**

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

MON

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

P2

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

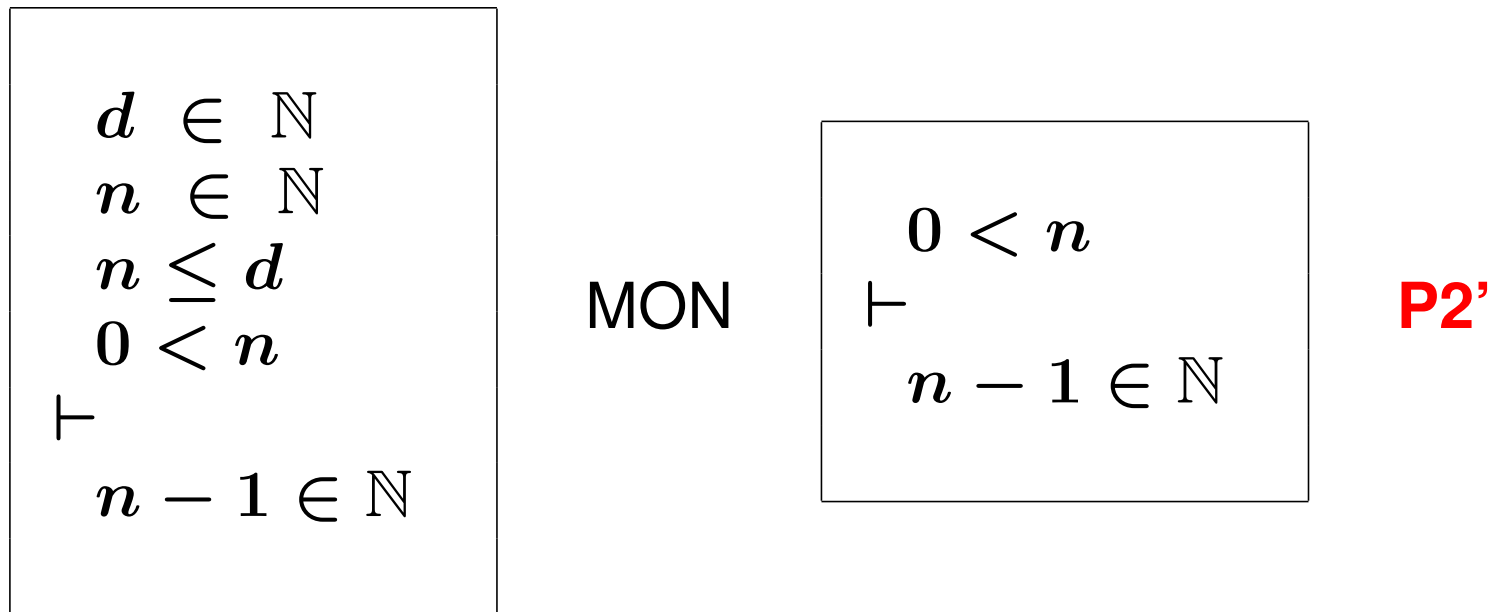
MON

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

INC

- We can conclude now with rule **INC**

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



- We can now conclude with rule **P2'**

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

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

MON

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

DEC

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

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

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

$$n \leq d$$

$$n < d$$

⊢

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

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

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

$$n \leq d$$

$$n < d$$

⊢

$$n + 1 \leq d$$

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

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

$$n \leq d$$

$$0 < n$$

⊢

$$n - 1 \in \mathbb{N}$$

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

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

$$n \leq d$$

$$0 < n$$

⊢

$$n - 1 \leq d$$

- Our system must be **initialized** (with no car in the island-bridge)
- The initialization event is **never guarded**
- It does **not mention any variable** in the right hand side of $:=$
- Its before-after predicate is just an **after predicate**

init

$n := 0$

After predicate

$n' = 0$

- Given c with axioms $A(c)$ and v with invariants $I(c, v)$
- Given an init event with after predicate $v' = K(c)$
- The Invariant Establishment Rule is the following:

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

axm0_1
⊢
Modified **inv0_1**

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

inv0_1 / INV

axm0_1
⊢
Modified **inv0_2**

$d \in \mathbb{N}$
⊢
 $0 \leq d$

inv0_2 / INV

- First Peano Axiom

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

- Third Peano Axiom (slightly modified)

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

$$\begin{array}{l} d \in \mathbb{N} \\ \vdash \\ \mathbf{0} \in \mathbb{N} \end{array}$$

MON

$$\begin{array}{l} \vdash \\ \mathbf{0} \in \mathbb{N} \end{array}$$

P1

$$\begin{array}{l} d \in \mathbb{N} \\ \vdash \\ \mathbf{0} \leq d \end{array}$$

P3

- It is possible for the system to be blocked if both guards are false
- We do not want this to happen
- We figure out that one important requirement was missing

Once started, the system should work for ever

FUN-4

- Given c with axioms $A(c)$ and v with invariants $I(c, v)$
- Given the guards $G_1(c, v), \dots, G_m(c, v)$ of the events
- We have to prove the following:

$\begin{array}{l} A(c) \\ I(c, v) \\ \vdash \\ G_1(c, v) \vee \dots \vee G_m(c, v) \end{array}$	DLF
---	--------------

$$\mathbf{axm0_1}$$
$$\mathbf{inv0_1}$$
$$\mathbf{inv0_2}$$
$$\vdash$$

Disjunction of guards

$$d \in \mathbb{N}$$
$$n \in \mathbb{N}$$
$$n \leq d$$
$$\vdash$$
$$n < d \vee 0 < n$$

- This cannot be proved **with the inference rules we have so far**
- $n \leq d$ can be replaced by $n = d \vee n < d$
- We have then to perform a **proof by cases** :
 - when $n = d$
 - when $n < d$

- **Proof by cases.** Note that we have **two antecedents**

$$\frac{\mathbf{H, P \vdash R} \quad \mathbf{H, Q \vdash R}}{\mathbf{H, P \vee Q \vdash R}} \quad \text{OR_L}$$

- Choice for proving a **disjunctive goal**

$$\frac{\mathbf{H \vdash P}}{\mathbf{H \vdash P \vee Q}} \quad \text{OR_R1}$$

$$\frac{\mathbf{H \vdash Q}}{\mathbf{H \vdash P \vee Q}} \quad \text{OR_R2}$$

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

MON

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

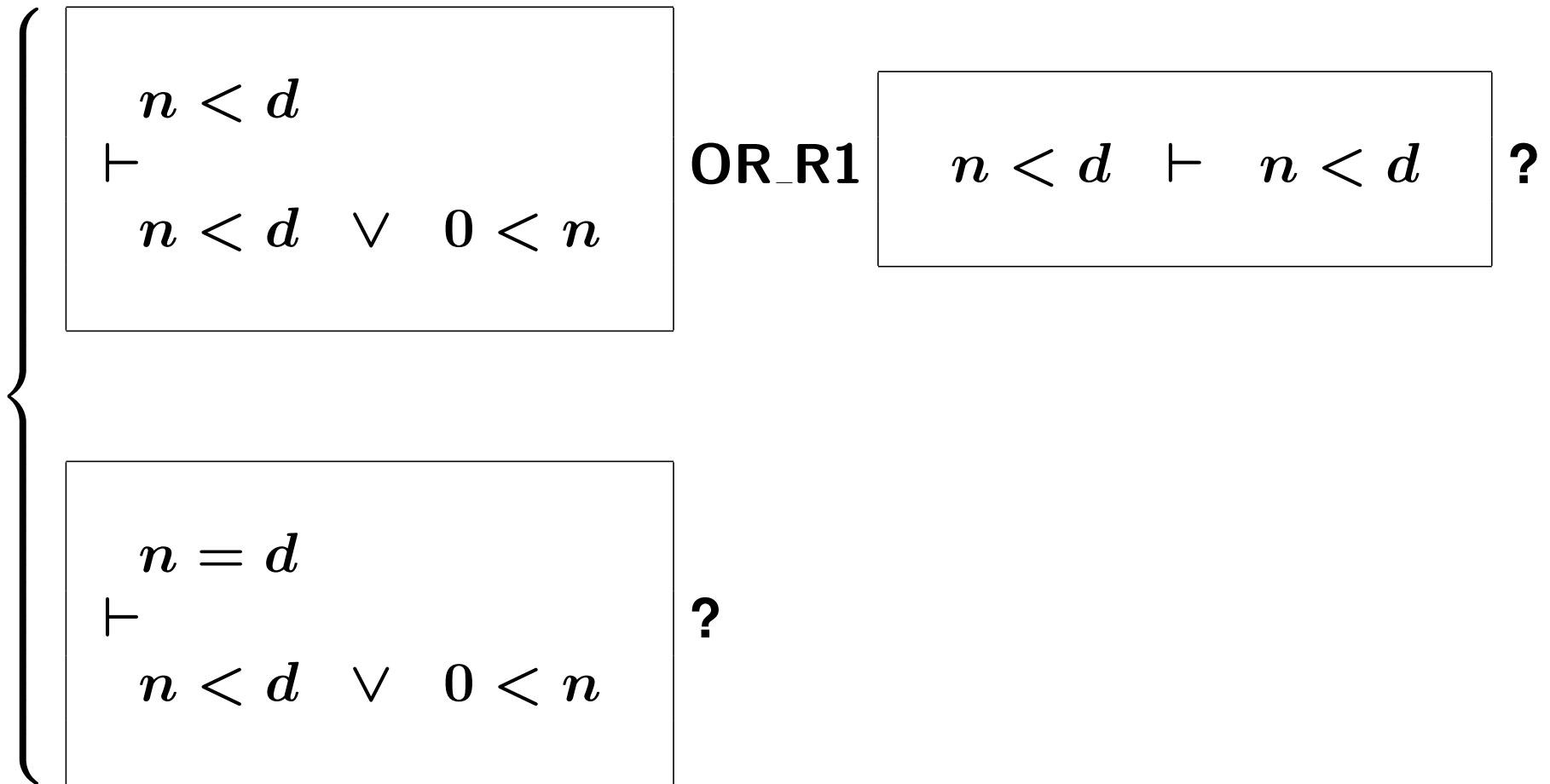
...

$$\begin{array}{l} n \leq d \\ \top \\ n < d \vee 0 < n \end{array}$$

OR_L

$$\begin{array}{l} n < d \\ \top \\ n < d \vee 0 < n \end{array} \dots$$

$$\begin{array}{l} n = d \\ \top \\ n < d \vee 0 < n \end{array} \dots$$



- The first ? seems to be **obvious**
- The second ? can be (partially) solved by **applying the equality**

- The **basic property** of a sequent

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

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

$$\frac{H_F, E = F \vdash P_F}{H_E, E = F \vdash P_E} \text{ EQ_LR}$$

$$\frac{H_E, E = F \vdash P_E}{H_F, E = F \vdash P_F} \text{ EQ_RL}$$

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

OR_R1

$$n < d \vdash n < d \quad \text{HYP}$$

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

EQ_LR

$$\vdash d < d \vee 0 < d \quad \text{OR_R2}$$

$$\text{OR_R2} \quad \vdash 0 < d \quad ?$$

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

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

$$\text{axm0_2: } 0 < d$$

- Thanks to the **proofs**, we discovered **3 errors**
- They were corrected by:
 - adding **guards** to both events
 - adding an **axiom**

- We have seen three Proof Rules:
 - The **Invariant Establishment** Rule: INV
 - The **Invariant Preservation** Rule: INV
 - The **Deadlock Freeness** Rule (not mandatory): DLF

<p>Axioms \vdash Modified Invariant</p>	<p>INV</p>
--	------------

<p>Axioms Invariants Guard of the event \vdash Modified Invariant</p>	<p>INV</p>
--	------------

<p>Axioms Invariants \vdash Disjunction of the guards</p>	<p>DLF</p>
---	------------

$$\frac{H1 \vdash G}{H1, H2 \vdash G} \text{ MON}$$

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

$$\frac{H, P \vdash R \quad H, Q \vdash R}{H, P \vee Q \vdash R} \text{ OR_L}$$

$$\frac{H \vdash P}{H \vdash P \vee Q} \text{ OR_R1}$$

$$\frac{H \vdash Q}{H \vdash P \vee Q} \text{ OR_R2}$$

$$\frac{\mathbf{H}_F, \mathbf{E} = \mathbf{F} \vdash \mathbf{P}_F}{\mathbf{H}_E, \mathbf{E} = \mathbf{F} \vdash \mathbf{P}_E} \quad \text{EQ_LR}$$

$$\frac{\mathbf{H}_E, \mathbf{E} = \mathbf{F} \vdash \mathbf{P}_E}{\mathbf{H}_F, \mathbf{E} = \mathbf{F} \vdash \mathbf{P}_F} \quad \text{EQ_RL}$$

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

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

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

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

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

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

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

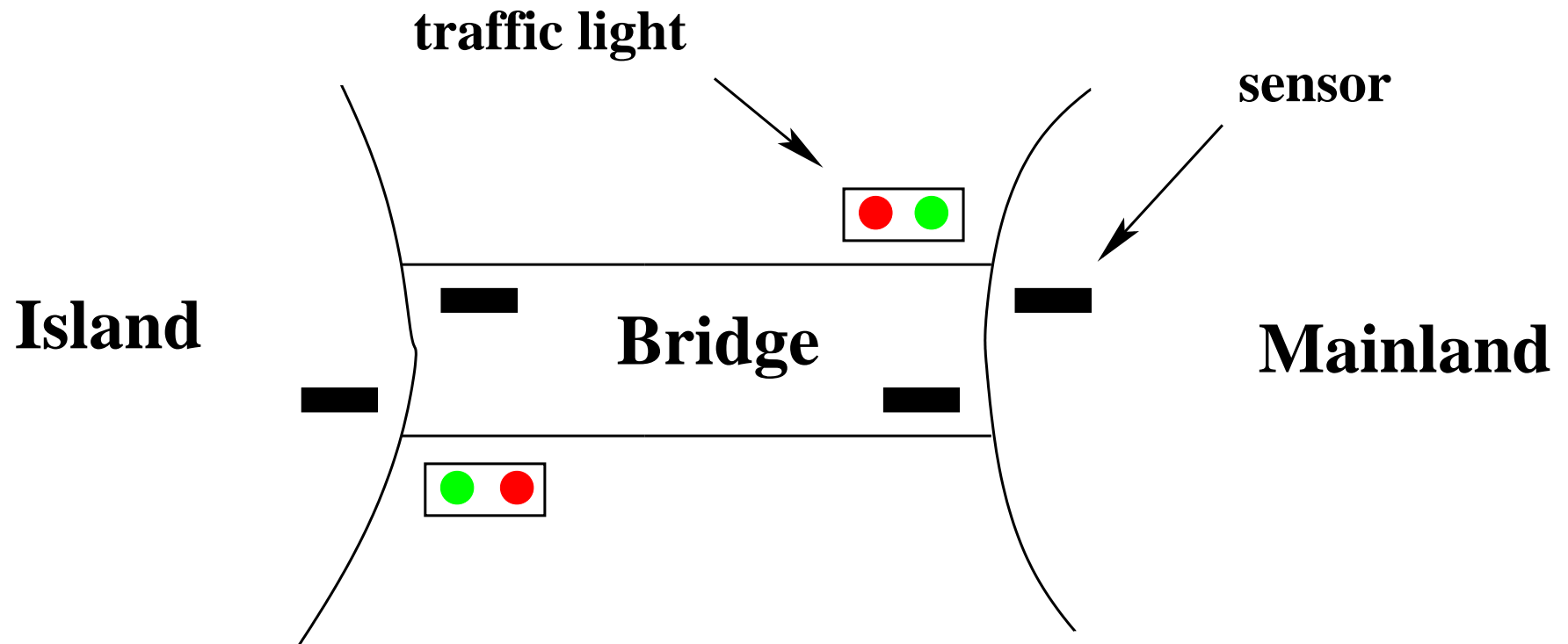
$0 < n$

then

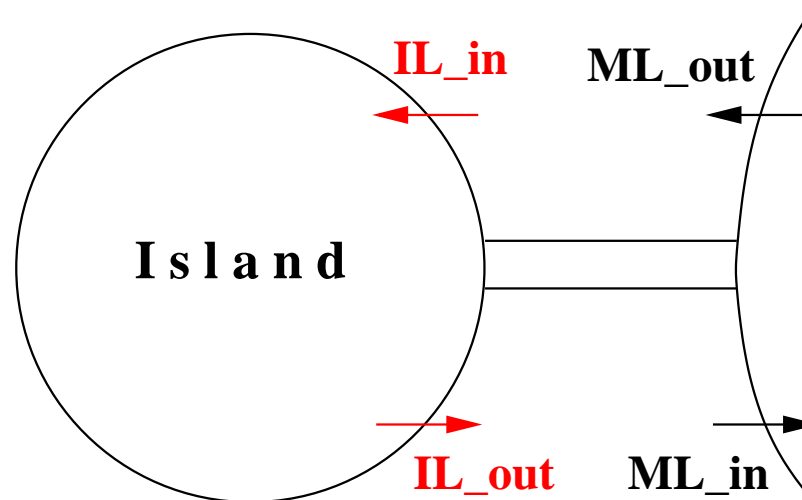
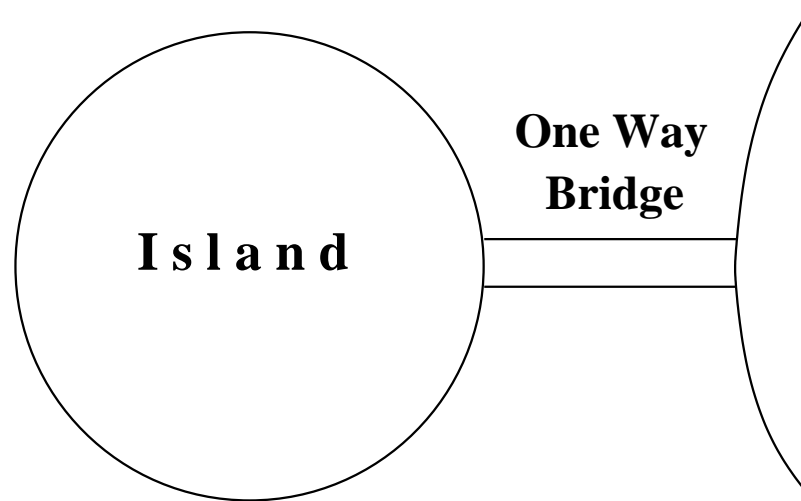
$n := n - 1$

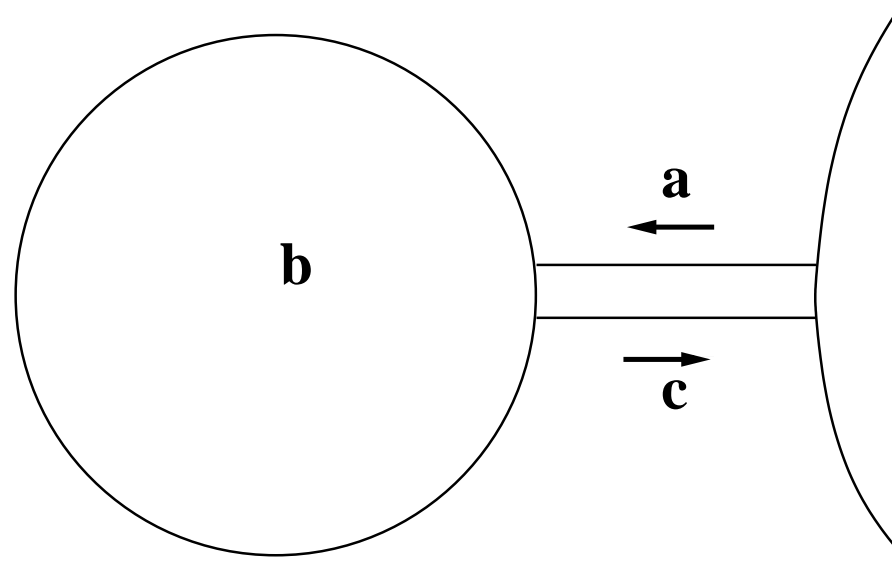
end

- **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)



- We go down with our **parachute**
- Our **view** of the system gets **more accurate**
- We introduce the **bridge** and **separate it from the island**
- We shall **refine** the state and the events
- But we add also **two new events: IL_in and IL_out**
- We are focusing on **FUN-3** (one way bridge)





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

- Variables a , b , and c denote **natural numbers**

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

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

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

- 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**)

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

constants: d

variables: a, b, c

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

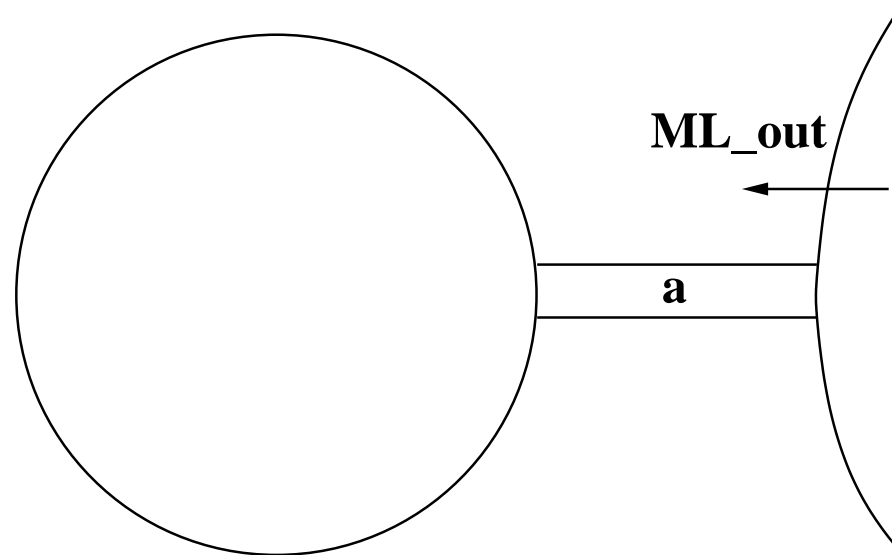
inv1_2: $b \in \mathbb{N}$

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

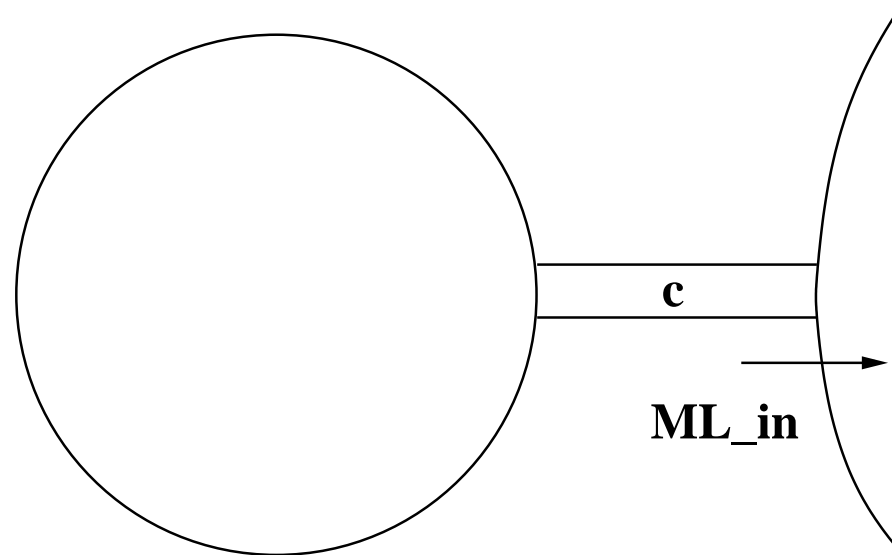
inv1_4: $a + b + c = n$

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

- 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



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



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

ML_out

when

$$a + b < d$$

$$c = 0$$

then

$$a := a + 1$$

end

ML_in

when

$$0 < c$$

then

$$c := c - 1$$

end

Before-after predicates showing the **unmodified variables**:

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

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

```
(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** seems to be **compatible**

```
(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**

- The concrete guard is stronger than the abstract one
- The actions are compatible

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 $G(c, v): G_1(c, v), G_2(c, v), \dots$

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

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

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
---	-----

- ML_out / GRD

- ML_in / GRD

axm0_1
axm0_2
inv0_1
inv0_2
inv1_1
inv1_2
inv1_3
inv1_4
inv1_5

Concrete guards of ML_out

⊢

Abstract guard of ML_out

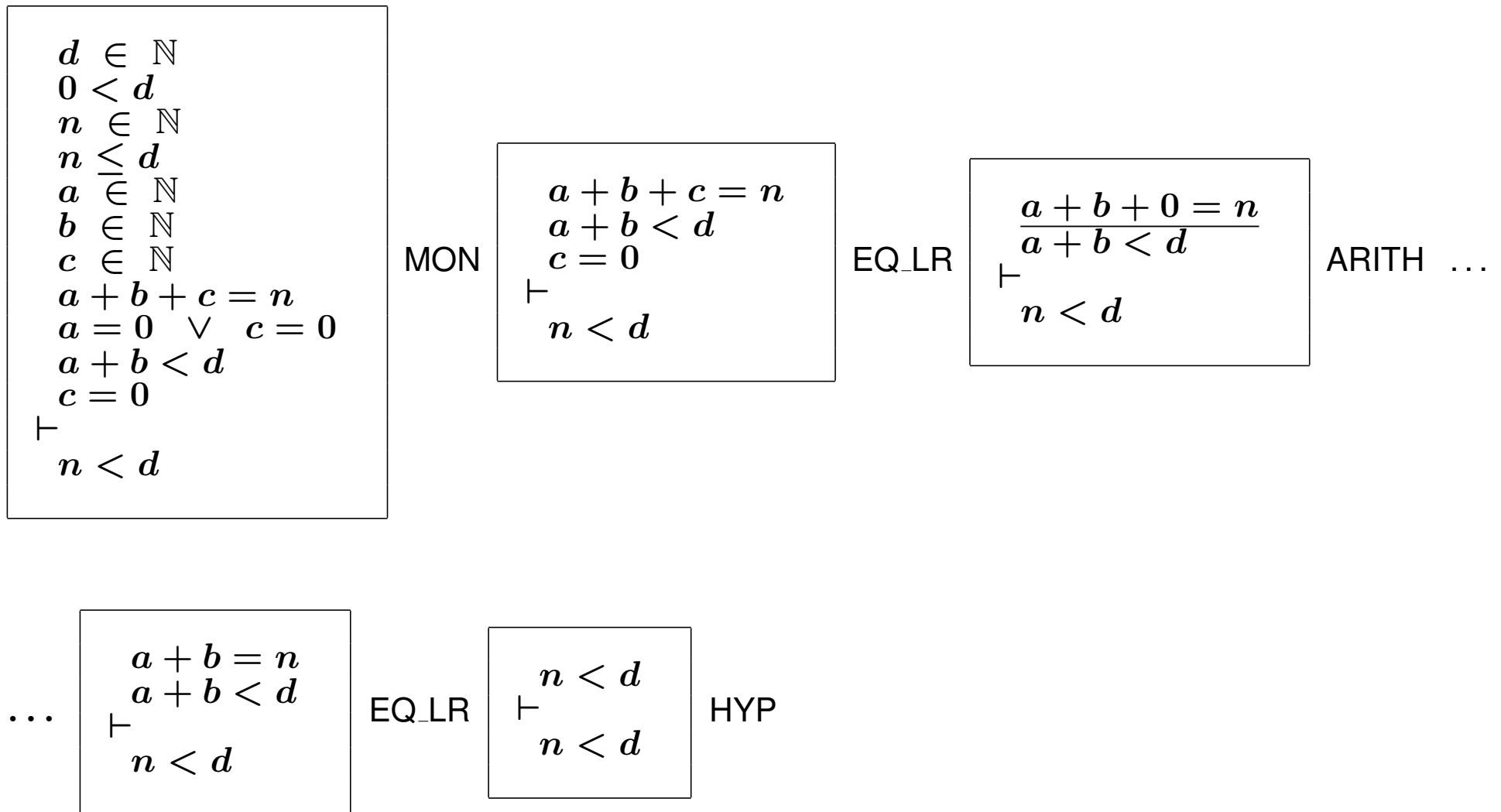
$d \in \mathbb{N}$
 $0 < d$
 $n \in \mathbb{N}$
 $n \leq d$
 $a \in \mathbb{N}$
 $b \in \mathbb{N}$
 $c \in \mathbb{N}$
 $a + b + c = n$
 $a = 0 \vee c = 0$
 $a + b < d$
 $c = 0$

⊢
 $n < d$

ML_out / GRD

(abstract-)ML_out
when
 $n < d$
then
 $n := n + 1$
end

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



- We use the "rule" ARITH to stand for **simple arithmetic simplifications** (where underlined)

axm0_1
axm0_2
inv0_1
inv0_2
inv1_1
inv1_2
inv1_3
inv1_4
inv1_5

Concrete guard of ML_in

⊢

Abstract guard of ML_in

$$\begin{aligned}
 & d \in \mathbb{N} \\
 & 0 < d \\
 & n \in \mathbb{N} \\
 & n \leq d \\
 & a \in \mathbb{N} \\
 & b \in \mathbb{N} \\
 & c \in \mathbb{N} \\
 & a + b + c = n \\
 & a = 0 \vee c = 0
 \end{aligned}$$

⊢

$$0 < n$$

ML_in / GRD

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

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

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

MON

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

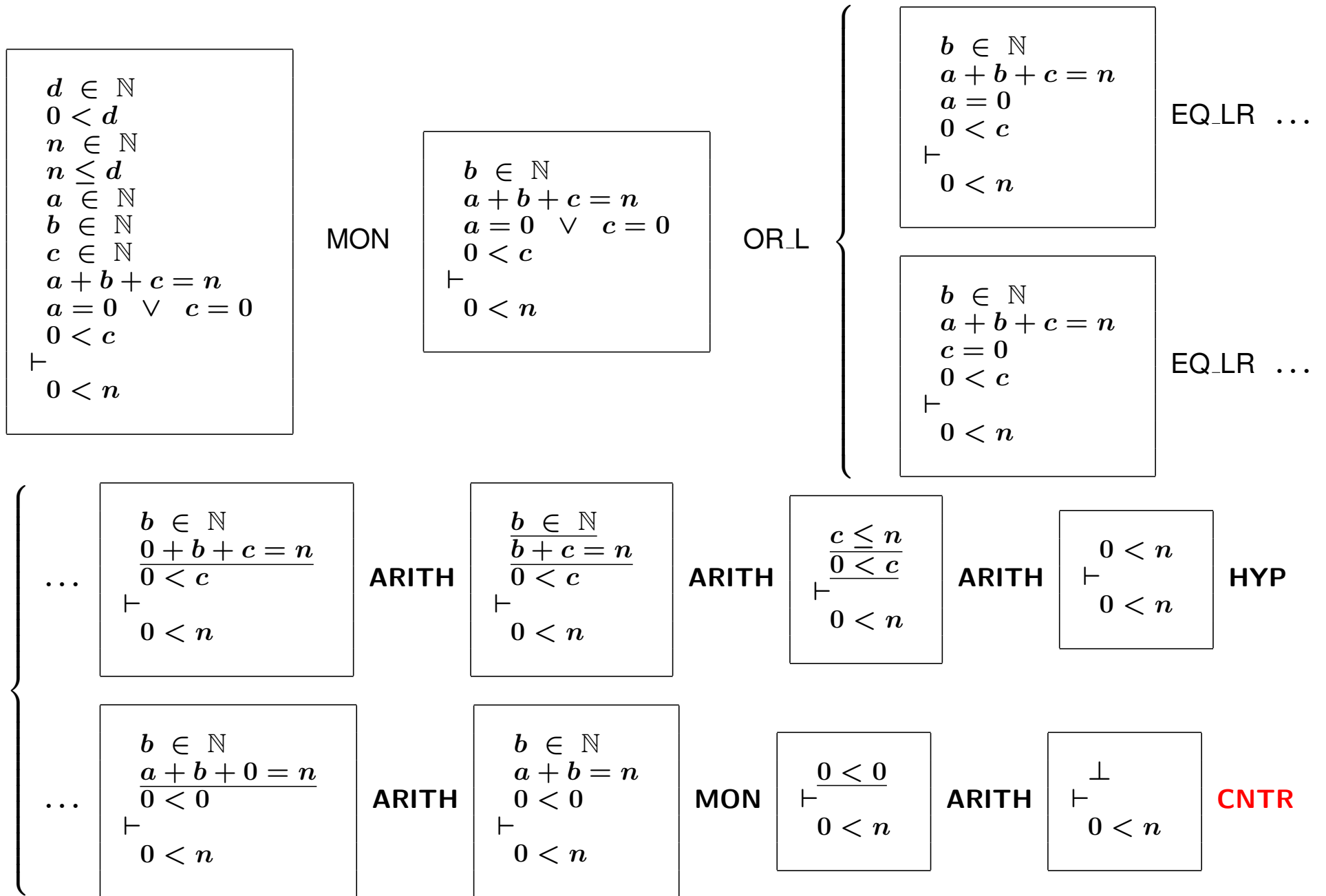
OR_L

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

EQ_LR ...

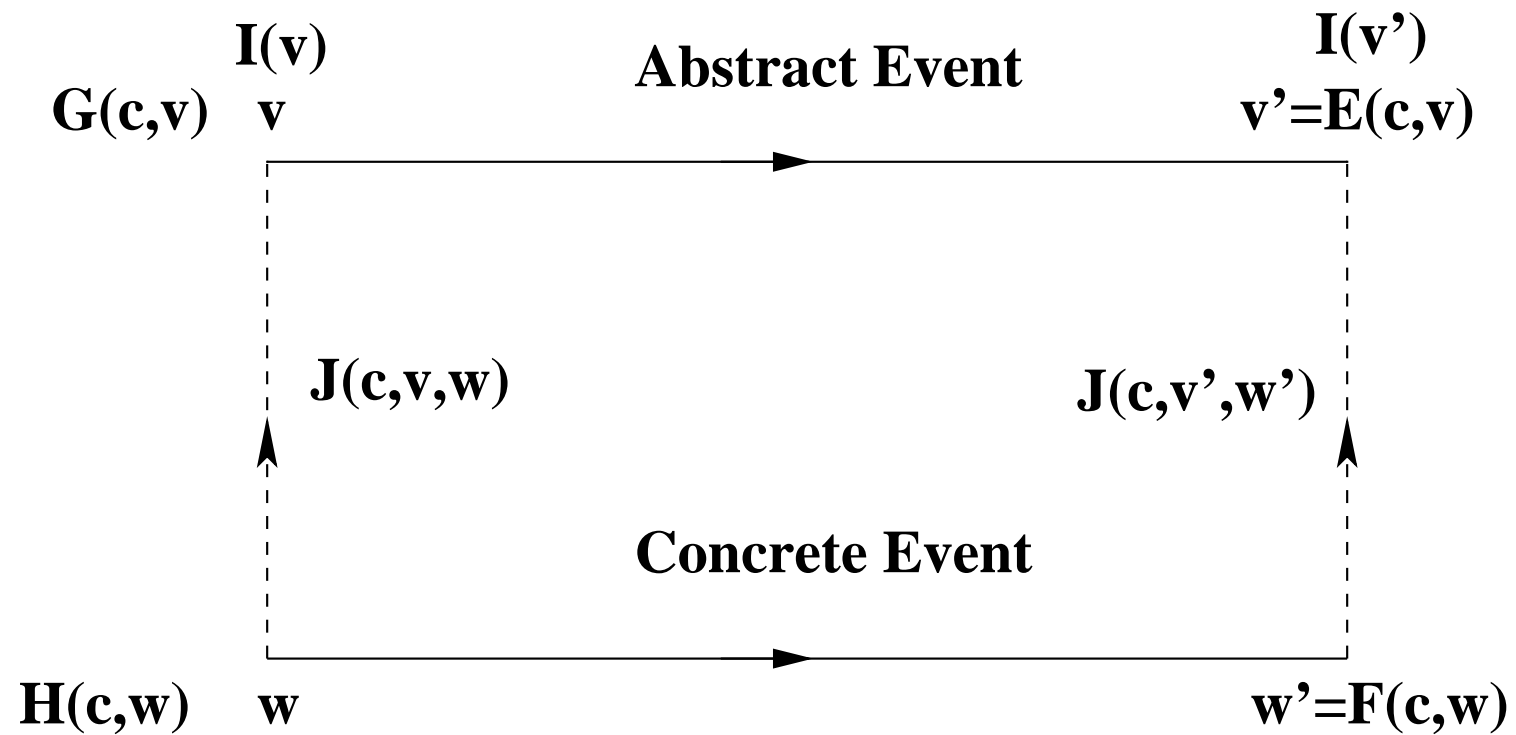
$$\begin{array}{l}
 b \in \mathbb{N} \\
 a + b + c = n \\
 c = 0 \\
 0 < c \\
 \top \\
 0 < n
 \end{array}$$

EQ_LR ...



- In the previous proof, we have used an additional inference rule
- It says that a **false hypothesis entails any goal**

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



<p>Axioms Abstract Invariants Concrete Invariants Concrete Guards \vdash Modified Concrete Invariant</p>	<p>$A(c)$ $I(c, v)$ $J(c, v, w)$ $H(c, w)$ \vdash $J_j(c, E(c, v), F(c, w))$</p>	<p>INV</p>
---	--	------------

- ML_out / GRD **done**
- ML_in / GRD **done**
- ML_out / **inv1_4** / INV
- ML_out / **inv1_5** / INV
- ML_in / **inv1_4** / INV
- ML_in / **inv1_5** / INV

axm0_1
 axm0_2
 inv0_1
 inv0_2
 inv1_1
 inv1_2
 inv1_3
 inv1_4
 inv1_5

Concrete guards of ML_out

⊢

Modified Invariant **inv1_4**

$$\begin{aligned}
 & d \in \mathbb{N} \\
 & 0 < d \\
 & n \in \mathbb{N} \\
 & n \leq d \\
 & a \in \mathbb{N} \\
 & b \in \mathbb{N} \\
 & c \in \mathbb{N} \\
 & \mathbf{a} + b + c = n \\
 & a = 0 \vee c = 0 \\
 & a + b < d \\
 & c = 0
 \end{aligned}$$

⊢

$$\mathbf{a} + \mathbf{1} + b + c = n + 1$$

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
 $\mathbf{a} := \mathbf{a} + \mathbf{1}$
end

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

MON

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

ARITH ...

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

EQ_LR

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

EQL

- In the previous proof, we have used an additional inference rule
- It says that a **any term is equal to itself**

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

- Remark: we have not yet formally defined "**term**" and "**predicate**"
- This will be done in later lectures

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_5**

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

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

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

MON

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

OR_R2

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

HYP

axm0_1
 axm0_2
 inv0_1
 inv0_2
 inv1_1
 inv1_2
 inv1_3
 inv1_4
 inv1_5

Concrete guards of ML_in

⊢

Modified Invariant **inv1_4**

$$\begin{aligned}
 & d \in \mathbb{N} \\
 & 0 < d \\
 & n \in \mathbb{N} \\
 & n \leq d \\
 & a \in \mathbb{N} \\
 & b \in \mathbb{N} \\
 & c \in \mathbb{N} \\
 & a + b + c = n \\
 & a = 0 \vee c = 0 \\
 & 0 < c
 \end{aligned}$$

⊢

$$a + b + c - 1 = n - 1$$

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

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

MON

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

EQ_LR

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

EQL

axm0_1
 axm0_2
 inv0_1
 inv0_2
 inv1_1
 inv1_2
 inv1_3
 inv1_4
 inv1_5

Concrete guards of ML_in

⊢

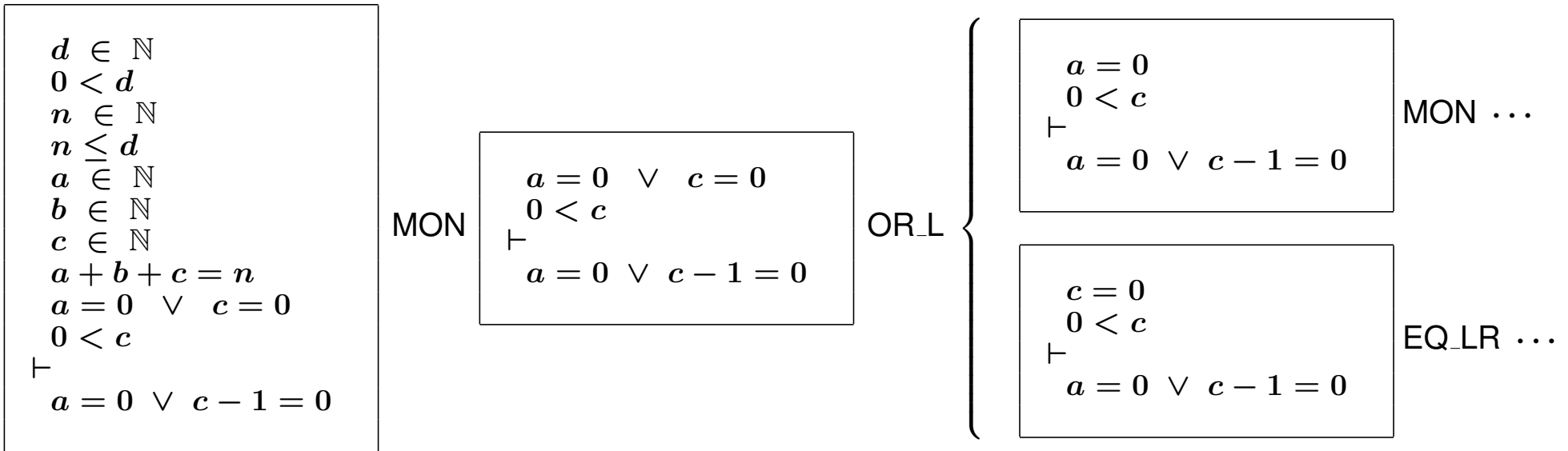
Modified Invariant **inv1_5**

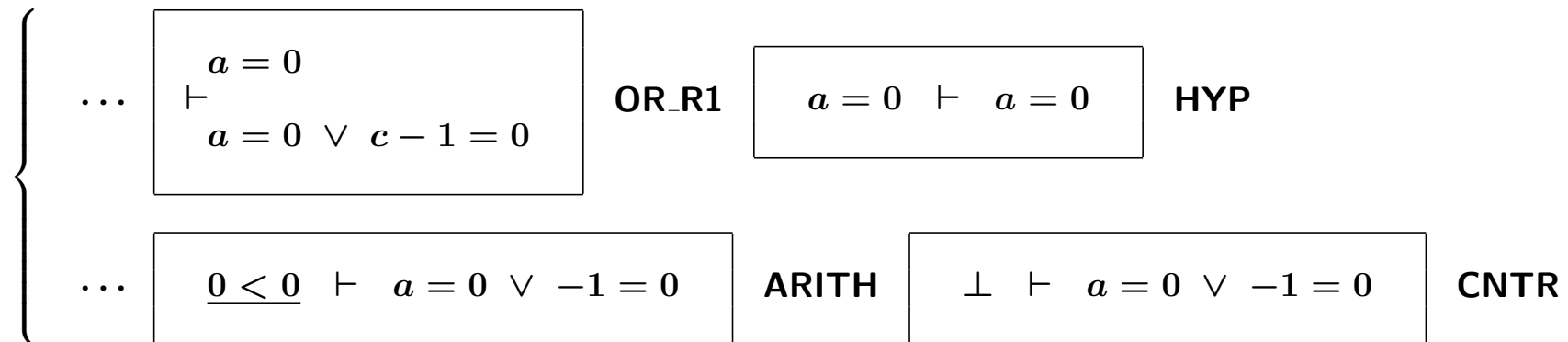
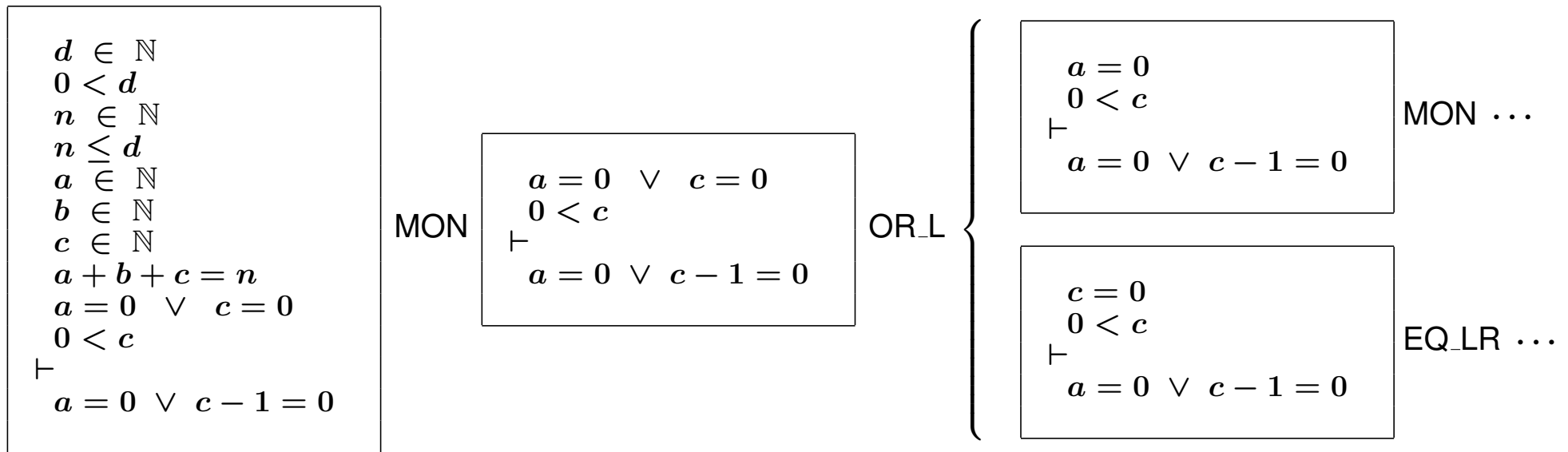
$$\begin{array}{l}
 d \in \mathbb{N} \\
 0 < d \\
 n \in \mathbb{N} \\
 n \leq d \\
 a \in \mathbb{N} \\
 b \in \mathbb{N} \\
 c \in \mathbb{N} \\
 a + b + c = n \\
 a = 0 \vee c = 0 \\
 0 < c \\
 \vdash \\
 a = 0 \vee c - 1 = 0
 \end{array}$$

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





- Concrete initialization

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

- Corresponding after predicate

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

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

⊢

Modified concrete invariants

$A(c)$

⊢

$J_j(c, K(c), L(c))$

INV

-
- 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

axm0_1**axm0_2**

⊢

Modified concrete invariant **inv1_4**
 $(a + b + c = n)$

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

⊢

 $0 + 0 + 0 = 0$ **axm0_1****axm0_2**

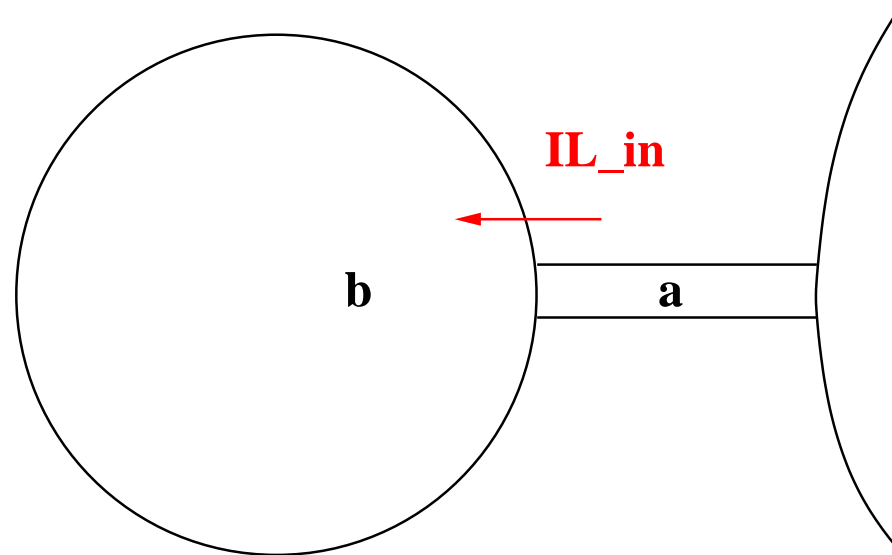
⊢

Modified concrete invariant **inv1_5**
 $(a = 0 \vee c = 0)$

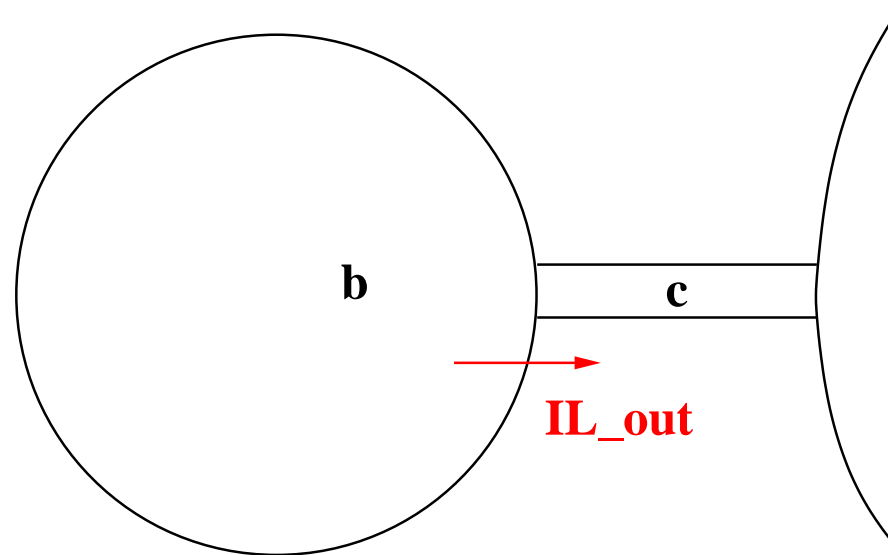
 $d \in \mathbb{N}$ $d > 0$

⊢

 $0 = 0 \vee 0 = 0$



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



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

Before-after predicates

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

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

The before-after predicate of **skip** in the **initial model**

$$n' = n$$

The before-after predicate of **skip** in the **first refinement**

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

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

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

-
- 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

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_4**

$d \in \mathbb{N}$
 $0 < d$
 $n \in \mathbb{N}$
 $n \leq d$
 $a \in \mathbb{N}$
 $b \in \mathbb{N}$
 $c \in \mathbb{N}$
 $a + b + c = n$
 $a = 0 \vee c = 0$
 $0 < a$

⊢

$a - 1 + b + 1 + c = n$

IL_in / **inv1_4** / INV

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

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

MON

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

ARITH

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

HYP

`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_5`

$$\begin{aligned}
 & d \in \mathbb{N} \\
 & 0 < d \\
 & n \in \mathbb{N} \\
 & n \leq d \\
 & a \in \mathbb{N} \\
 & b \in \mathbb{N} \\
 & c \in \mathbb{N} \\
 & a + b + c = n \\
 & a = 0 \vee c = 0 \\
 & 0 < a
 \end{aligned}$$

$$\vdash \quad a - 1 = 0 \vee c = 0$$

`IL_in` / `inv1_5` / `INV`

```

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

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

MON

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

OR_L ...

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

MON

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

OR_L ...

$$\left. \begin{array}{l} \dots \\ \dots \end{array} \right\} \begin{array}{l}
 a = 0 \\
 0 < a \\
 \vdash \\
 a - 1 = 0 \vee c = 0
 \end{array}$$

EQ_LR

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

ARITH

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

CNTR

$$\left. \begin{array}{l} \dots \\ \dots \end{array} \right\} \begin{array}{l}
 c = 0 \\
 0 < a \\
 \vdash \\
 a - 1 = 0 \vee c = 0
 \end{array}$$

MON

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

OR_R2

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

HYP

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
--	-----

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 invariant

Concrete invariant

Concrete guard

⊢

Modified Var. < Var.

$A(c)$

$I(c, v)$

$J(c, v, w)$

$H(c, w)$

⊢

$V(c, F(c, w)) < V(c, w)$

VAR

variant_1: $2 * a + b$

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

- NAT
- IL_in / VAR
- IL_out / VAR

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

$d \in \mathbb{N}$
 $0 < d$
 $n \in \mathbb{N}$
 $n \leq d$
 $a \in \mathbb{N}$
 $b \in \mathbb{N}$
 $c \in \mathbb{N}$
 $a + b + c = n$
 $a = 0 \vee c = 0$
 $0 < a$

⊢
 $2 * (a - 1) + b + 1 < 2 * a + b$

IL_in / VAR

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

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

$$\begin{aligned} & d \in \mathbb{N} \\ & 0 < d \\ & n \in \mathbb{N} \\ & n \leq d \\ & a \in \mathbb{N} \\ & b \in \mathbb{N} \\ & c \in \mathbb{N} \\ & a + b + c = n \\ & a = 0 \vee c = 0 \\ & 0 < b \\ & a = 0 \end{aligned}$$

⊢
 $2 * a + b - 1 < 2 * a + b$

IL_out / VAR

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

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

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

We have to prove the following Deadlock Freedom Rule

$\begin{array}{l} A(c) \\ I(c, v) \\ J(c, v, w) \\ G_1(c, v) \vee \dots \vee G_m(c, v) \\ \vdash \\ H_1(c, w) \vee \dots \vee H_n(c, w) \end{array}$	DLF
--	-----

axm0_1
axm0_2
inv0_1
inv0_2
inv1_1
inv1_2
inv1_3
inv1_4
inv1_5

\vdash
 Disjunction of concrete guards

$d \in \mathbb{N}$
 $0 < d$
 $n \in \mathbb{N}$
 $n \leq d$
 $a \in \mathbb{N}$
 $b \in \mathbb{N}$
 $c \in \mathbb{N}$
 $a + b + c = n$
 $a = 0 \vee c = 0$

DLF

\vdash
 $(a + b < d \wedge c = 0) \vee$
 $c > 0 \vee a > 0$
 $(b > 0 \wedge a = 0)$

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

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

$$\frac{H, P, Q \vdash R}{H, P \wedge Q \vdash R} \text{ AND_L}$$

$$\frac{H \vdash P \quad H \vdash Q}{H \vdash P \wedge Q} \text{ AND_R}$$

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

MON

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

NEG

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

ARITH...

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

MON

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

NEG

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

ARITH...

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

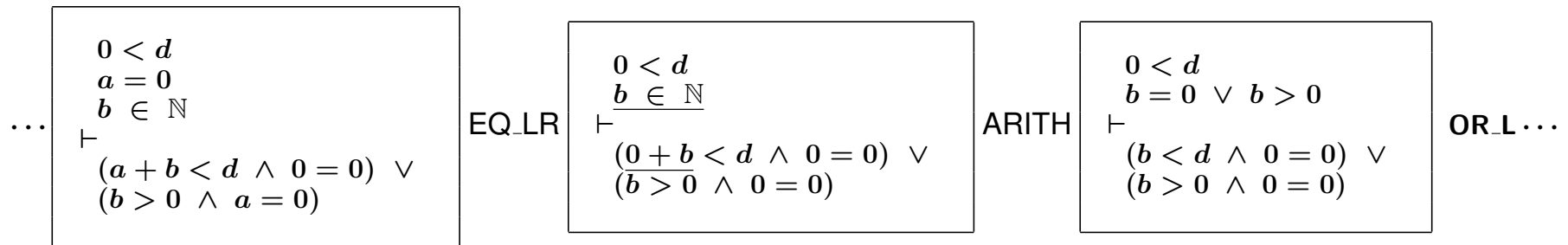
EQ_LR

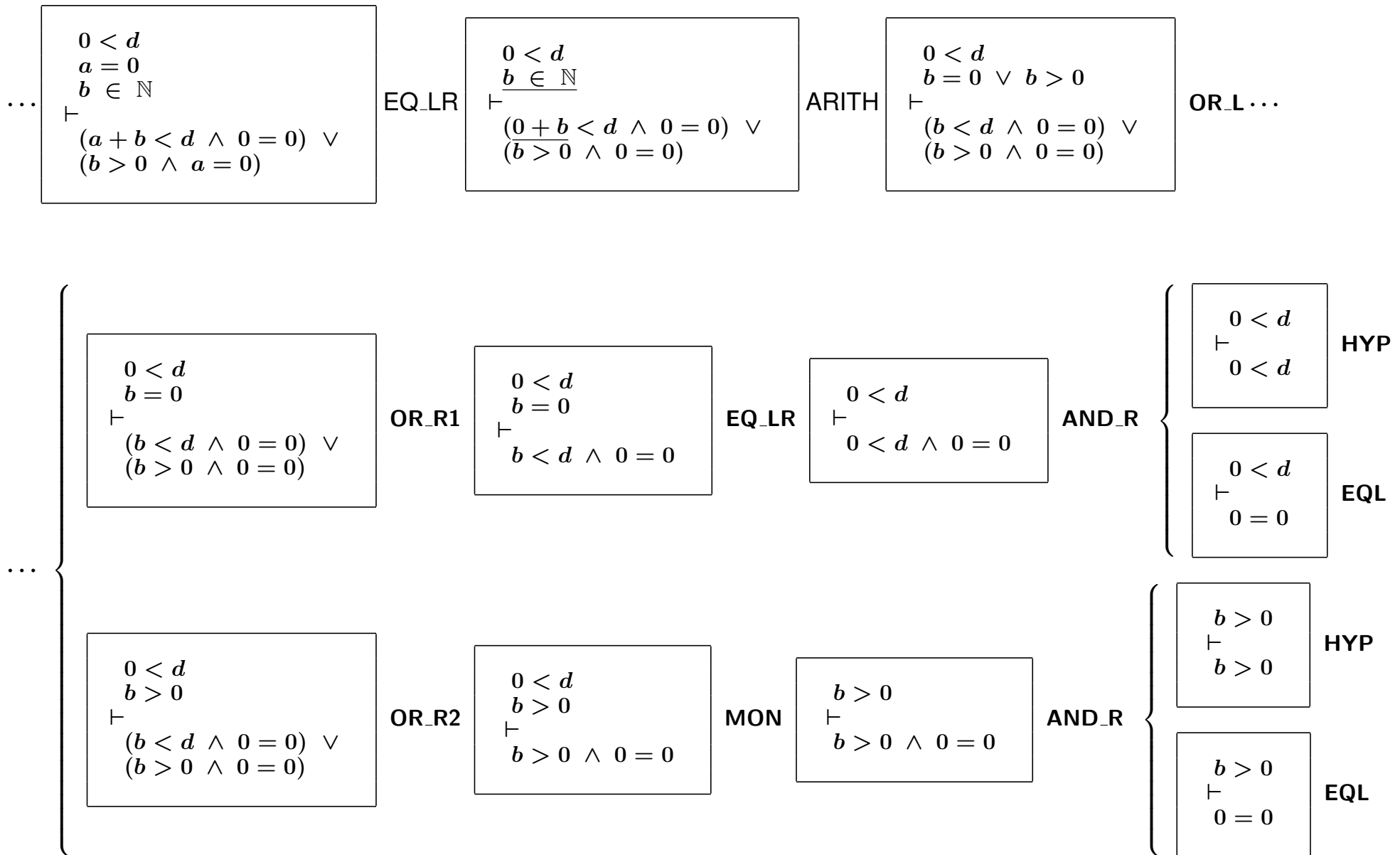
$$\begin{array}{l}
 0 < d \\
 a \in \mathbb{N} \\
 b \in \mathbb{N} \\
 \vdash \\
 (a + b < d \wedge 0 = 0) \vee \\
 a > 0 \vee \\
 (b > 0 \wedge a = 0)
 \end{array}$$

NEG

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

ARITH...





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

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

- For old events:
 - Strengthening of guards: rule **GRD**
 - Concrete invariant preservation: rule **INV**
- For new events:
 - Refining the implicit **skip event**: rule **INV**
 - Absence of divergence: rules **NAT** and **VAR**
- For all events:
 - Relative deadlock freeness: rule **DLF**

Axioms Abstract invariants Concrete invariants Concrete guards ┆ Abstract guard	GRD
--	-----

Axioms Abstract invariants Concrete invariants Concrete guard ┆ Modified concrete invariant	INV
--	-----

Axioms ┆ Modified concrete invariant	INV
--	-----

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
--	-----

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: $a = 0 \vee c = 0$

variant1: $2 * a + b$

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

$0 < b$

$a = 0$

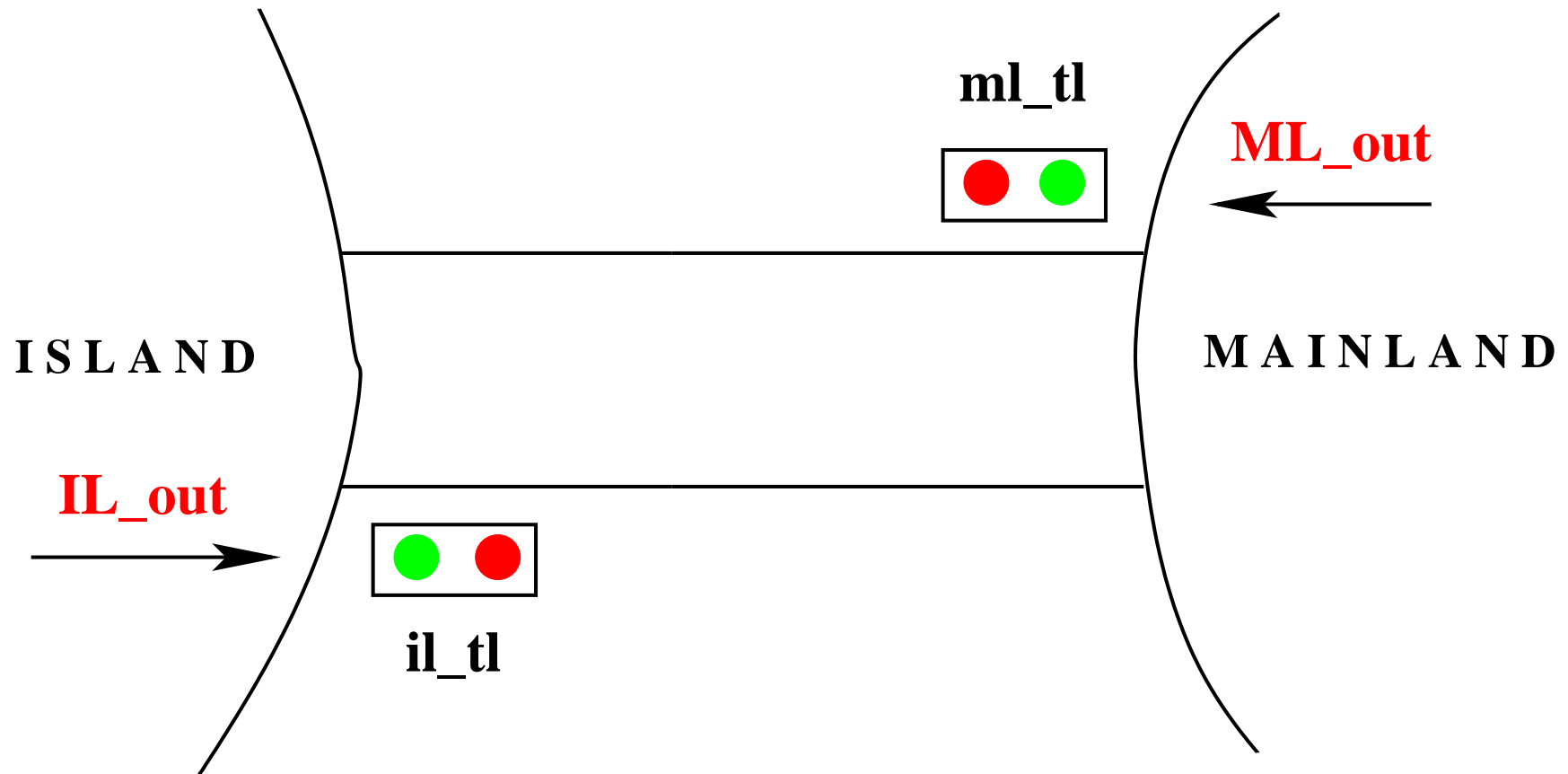
then

$b := b - 1$

$c := c + 1$

end

- **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)



set: *COLOR*

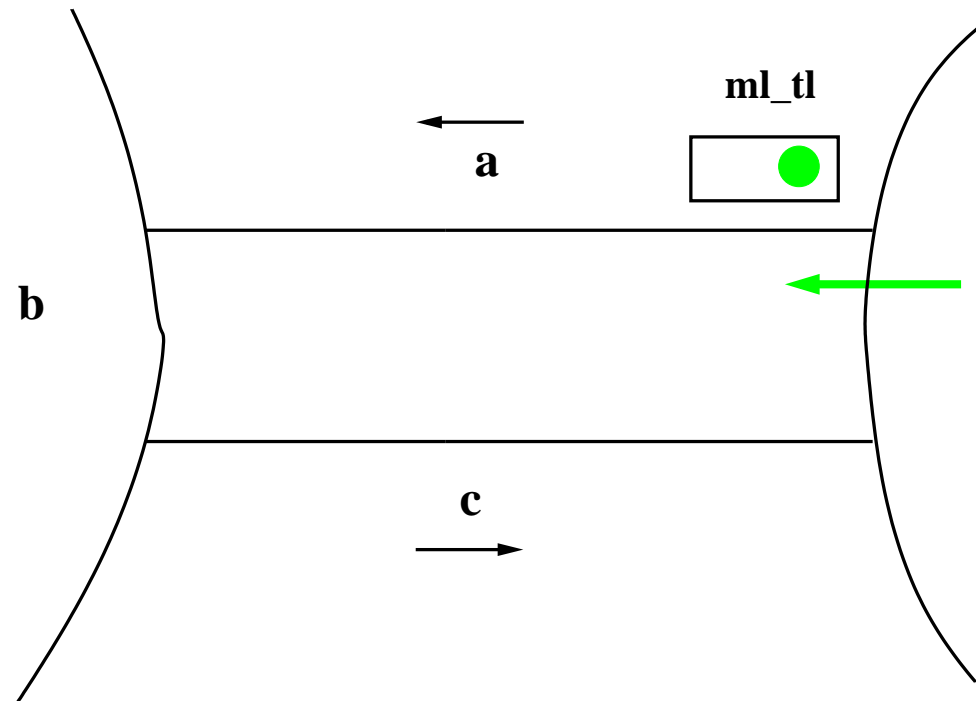
constants: *red, green*

axm2_1: *COLOR = {green, red}*

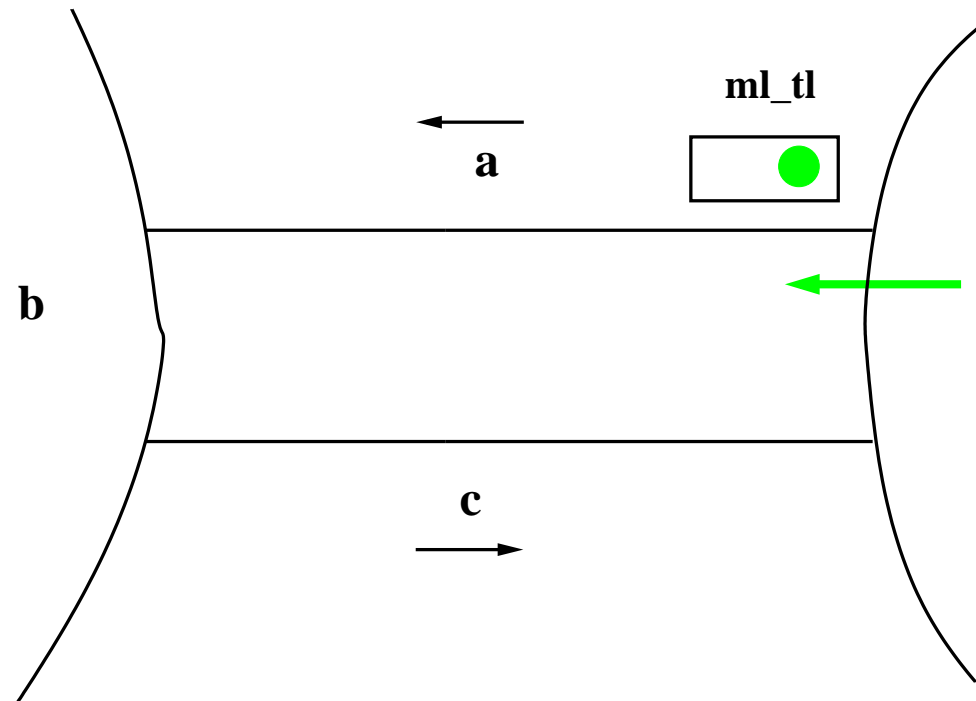
axm2_2: *green \neq red*

$$il_tl \in COLOR$$
$$ml_tl \in COLOR$$

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

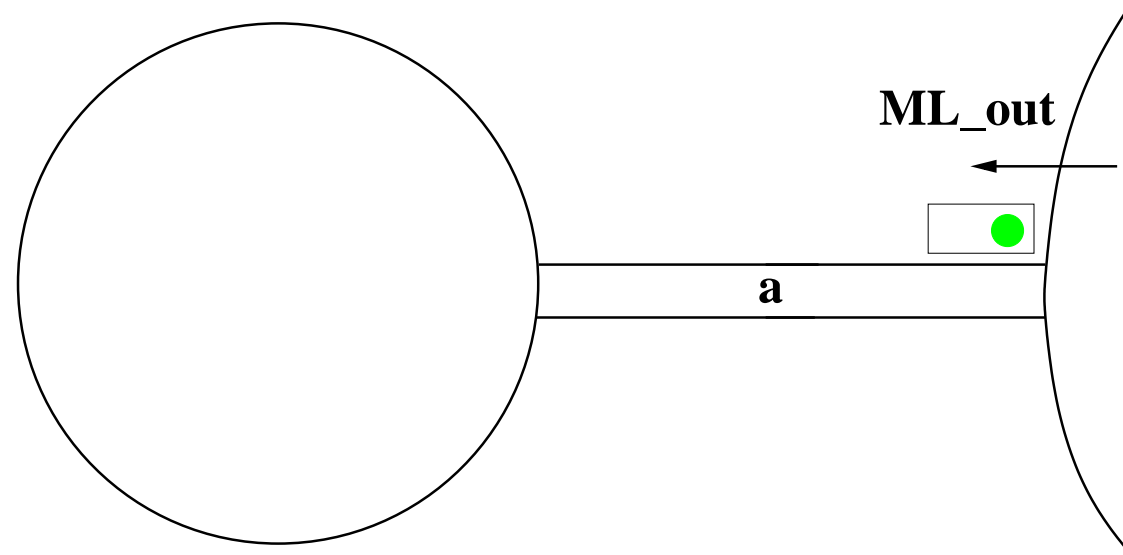


- A green **mainland traffic light** implies **safe access** to the bridge

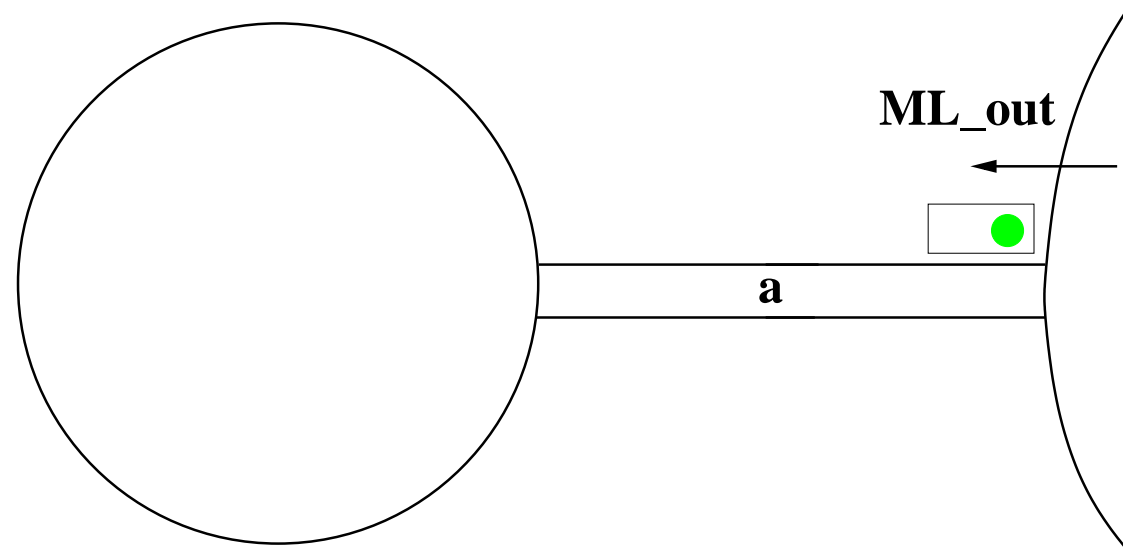


- A green **mainland traffic light** implies **safe access** to the bridge

$$ml_tl = \text{green} \Rightarrow c = 0 \wedge a + b < d$$

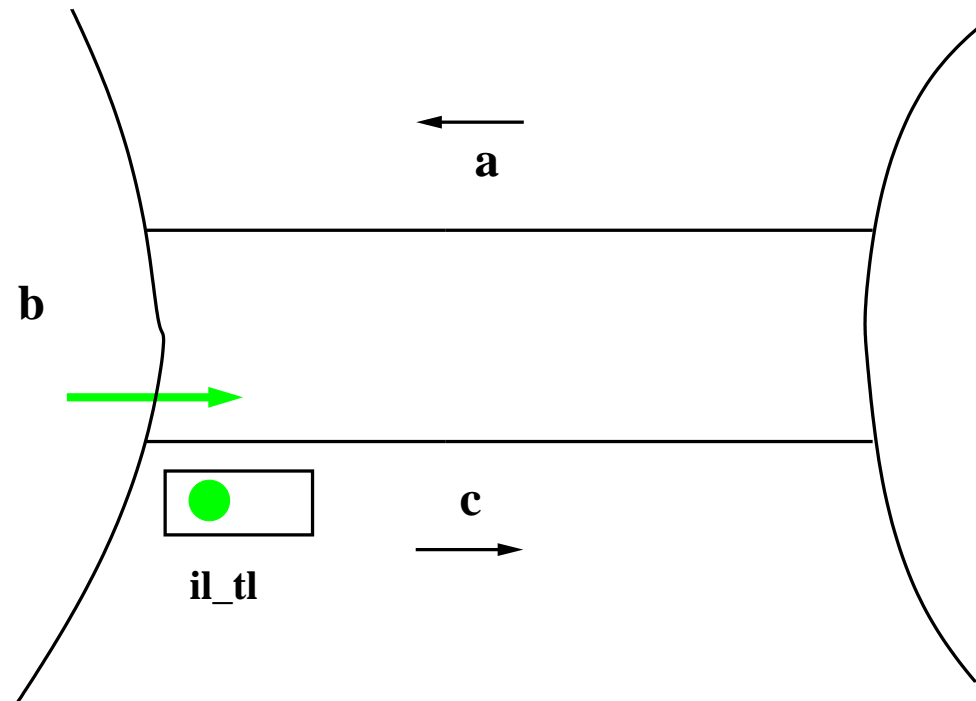


```
(abstract_)ML_out  
when  
   $c = 0$   
   $a + b < d$   
then  
   $a := a + 1$   
end
```

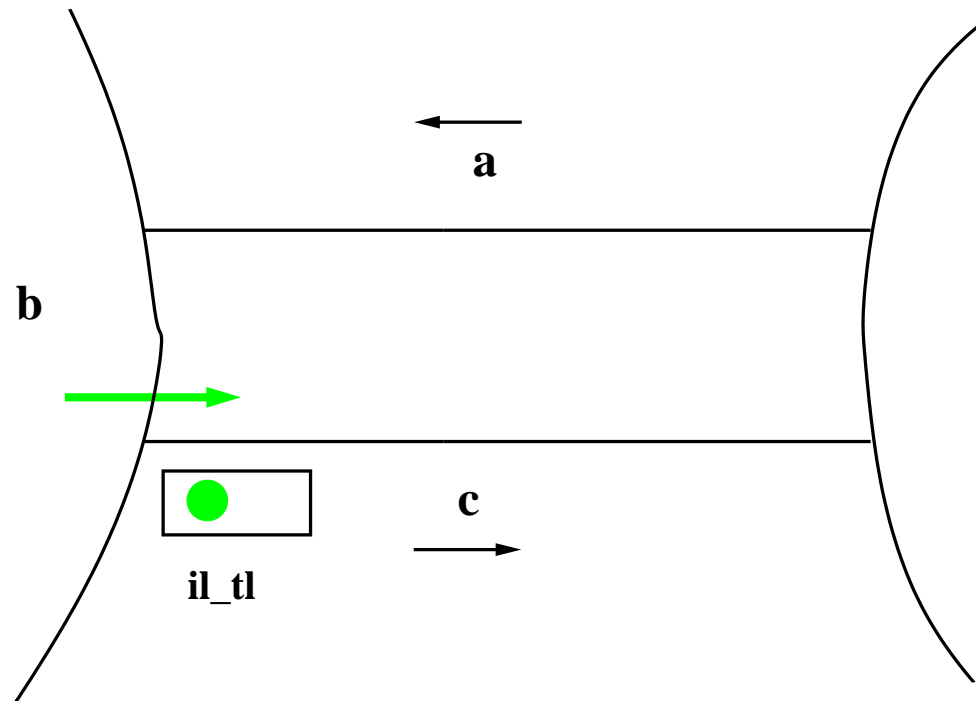


```
(abstract_)ML_out  
when  
   $c = 0$   
   $a + b < d$   
then  
   $a := a + 1$   
end
```

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

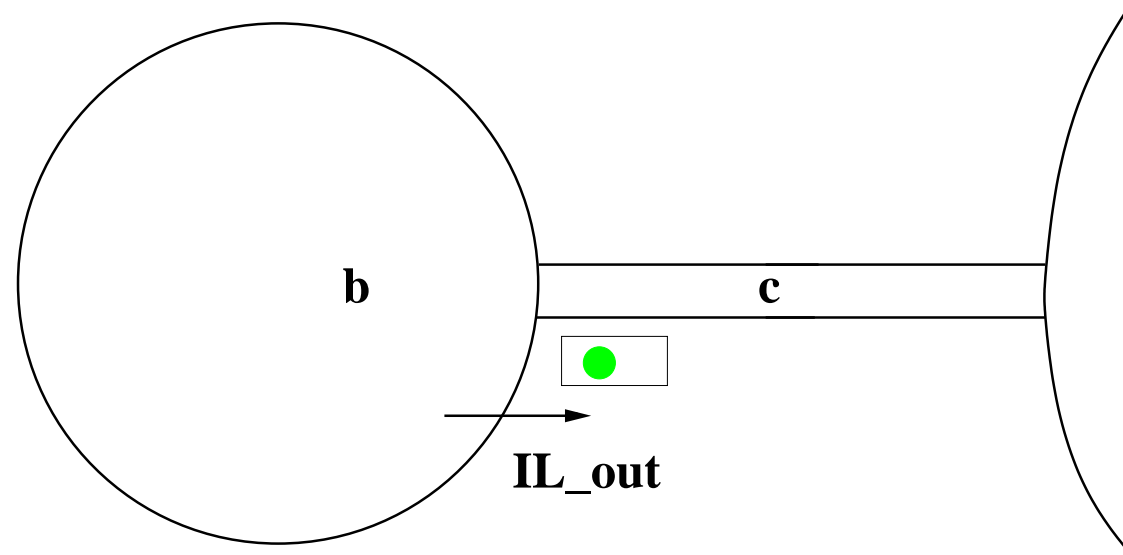


- A green **island traffic light** implies **safe access** to the bridge

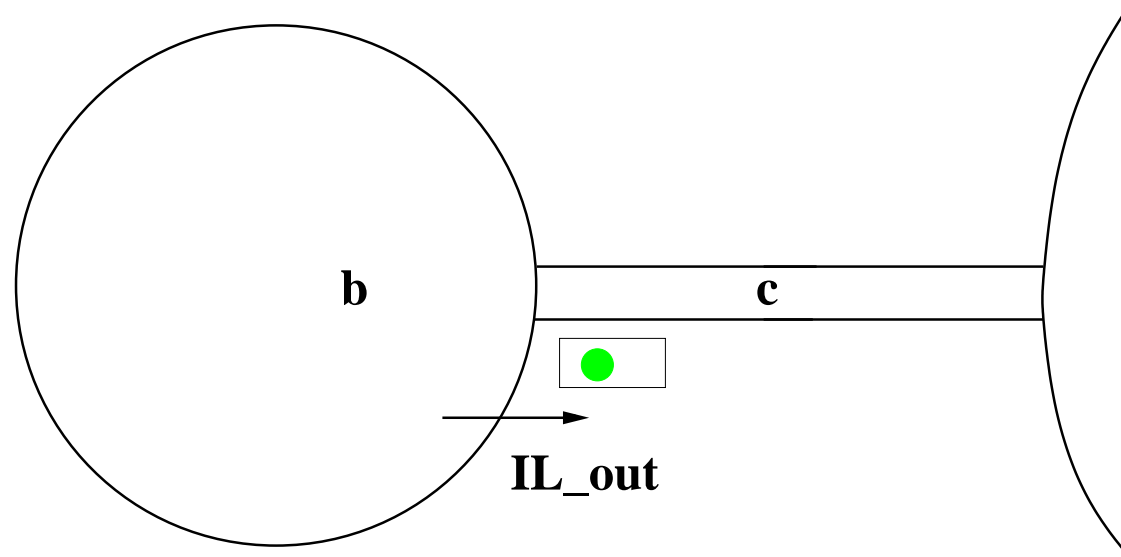


- A green **island traffic light** implies **safe access** to the bridge

$$il_tl = \text{green} \Rightarrow a = 0 \wedge 0 < b$$



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



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

```
(concrete_)IL_out  
when  
   $il\_tl = \text{green}$   
then  
   $b, c := b - 1, c + 1$   
end
```

ML_tl_green

when

$ml_tl = \text{red}$

$c = 0$

$a + b < d$

then

$ml_tl := \text{green}$

end

IL_tl_green

when

$il_tl = \text{red}$

$a = 0$

$0 < b$

then

$il_tl := \text{green}$

end

- Turning lights to **green** when **proper conditions hold**

variables: a, b, c, ml_tl, il_tl

inv2_1: $ml_tl \in COLOR$

inv2_2: $il_tl \in COLOR$

inv2_3: $ml_tl = green \Rightarrow a + b < d \wedge c = 0$

inv2_4: $il_tl = green \Rightarrow 0 < b \wedge a = 0$

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

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

Events ML_in and IL_in are unchanged

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

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

variables: a, b, c, ml_tl, il_tl

- Variables $a, b,$ and c were present in the previous refinement
- Variables ml_tl and il_tl are superposed to $a, b,$ and c
- We have thus to extend rule INV

Abstract_Event

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

- 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:

$$\mathbf{inv2_3:} \quad ml_tl = \mathbf{green} \Rightarrow a + b < d \wedge c = 0$$

$$\mathbf{inv2_4:} \quad il_tl = \mathbf{green} \Rightarrow 0 < b \wedge a = 0$$

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

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

- SIM is completely trivial since the actions are the same

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

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

- GRD is also trivial

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

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

```

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

```

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

```

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

```

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

```

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
    
```

```

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

```

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

- INV applied to ML_in and IL_in holds trivially

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

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

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

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

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

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

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

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

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

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

- INV applied to ML_out and IL_out **raise some difficulties**

- ML_out / **inv2_4** / INV

- IL_out / **inv2_3** / INV

- ML_out / **inv2_3** / INV

- IL_out / **inv2_4** / INV

- 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**

$$\frac{H \vdash P}{H, \neg P \vdash Q} \quad \text{NOT_L}$$

$$\frac{H, P \vdash Q \quad H, P \vdash \neg Q}{H \vdash \neg P} \quad \text{NOT_R}$$

axm0_1
 axm0_2
 axm2_1
 axm2_2
 inv0_1
 inv0_2
 inv1_1
 inv1_2
 inv1_3
 inv1_4
 inv1_5
 inv2_1
 inv2_2
 inv2_3
 inv2_4
 Guard of event ML_out
 \vdash
 Modified invariant **inv2_4**

$d \in \mathbb{N}$
 $0 < d$
 $COLOR = \{\text{green}, \text{red}\}$
 $\text{green} \neq \text{red}$
 $n \in \mathbb{N}$
 $n \leq d$
 $a \in \mathbb{N}$
 $b \in \mathbb{N}$
 $c \in \mathbb{N}$
 $a + b + c = n$
 $a = 0 \vee c = 0$
 $ml_tl \in COLOR$
 $il_tl \in COLOR$
 $ml_tl = \text{green} \Rightarrow a + b < d \wedge c = 0$
 $il_tl = \text{green} \Rightarrow 0 < b \wedge a = 0$
 $ml_tl = \text{green}$
 \vdash
 $il_tl = \text{green} \Rightarrow 0 < b \wedge a + 1 = 0$

ML_out / **inv2_4** / INV

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

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

MON

$$\begin{array}{l}
 \mathit{green} \neq \mathit{red} \\
 il_tl = \mathit{green} \Rightarrow 0 < b \wedge a = 0 \\
 ml_tl = \mathit{green} \\
 \vdash \\
 il_tl = \mathit{green} \Rightarrow 0 < b \wedge a + 1 = 0
 \end{array}$$

IMP_R...

$d \in \mathbb{N}$
 $0 < d$
 $COLOR = \{\text{green}, \text{red}\}$
 $\text{green} \neq \text{red}$
 $n \in \mathbb{N}$
 $n \leq d$
 $a \in \mathbb{N}$
 $b \in \mathbb{N}$
 $c \in \mathbb{N}$
 $a + b + c = n$
 $a = 0 \vee c = 0$
 $ml_tl \in COLOR$
 $il_tl \in COLOR$
 $ml_tl = \text{green} \Rightarrow a + b < d \wedge c = 0$
 $il_tl = \text{green} \Rightarrow 0 < b \wedge a = 0$
 $ml_tl = \text{green}$
 \vdash
 $il_tl = \text{green} \Rightarrow 0 < b \wedge a + 1 = 0$

MON

$\text{green} \neq \text{red}$
 $il_tl = \text{green} \Rightarrow 0 < b \wedge a = 0$
 $ml_tl = \text{green}$
 \vdash
 $il_tl = \text{green} \Rightarrow 0 < b \wedge a + 1 = 0$

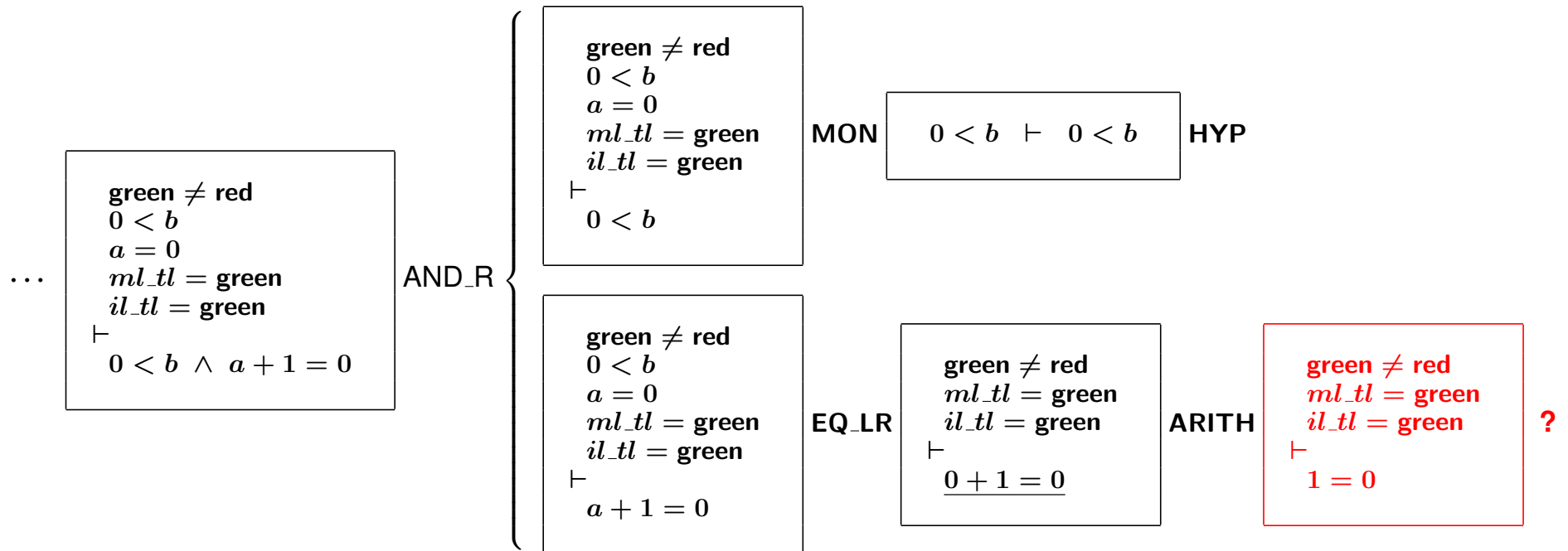
IMP_R ...

\dots
 $\text{green} \neq \text{red}$
 $il_tl = \text{green} \Rightarrow 0 < b \wedge a = 0$
 $ml_tl = \text{green}$
 $il_tl = \text{green}$
 \vdash
 $0 < b \wedge a + 1 = 0$

IMP_L

$\text{green} \neq \text{red}$
 $0 < b \wedge a = 0$
 $ml_tl = \text{green}$
 $il_tl = \text{green}$
 \vdash
 $0 < b \wedge a + 1 = 0$

AND_L ...



axm0_1
 axm0_2
 axm2_1
 axm2_2
 inv0_1
 inv0_2
 inv1_1
 inv1_2
 inv1_3
 inv1_4
 inv1_5
 inv2_1
 inv2_2
 inv2_3
 inv2_4
 Guard of IL_out
 \vdash
 Modified inv2_3

$d \in \mathbb{N}$
 $0 < d$
 $COLOR = \{\text{green}, \text{red}\}$
 $\text{green} \neq \text{red}$
 $n \in \mathbb{N}$
 $n \leq d$
 $a \in \mathbb{N}$
 $b \in \mathbb{N}$
 $c \in \mathbb{N}$
 $a + b + c = n$
 $a = 0 \vee c = 0$
 $ml_tl \in COLOR$
 $il_tl \in COLOR$
 $ml_tl = \text{green} \Rightarrow a + b < d \wedge c = 0$
 $il_tl = \text{green} \Rightarrow 0 < b \wedge a = 0$
 $il_tl = \text{green}$
 \vdash
 $ml_tl = \text{green} \Rightarrow a + b - 1 < d \wedge c + 1 = 0$

IL_out / inv2_3 / INV

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

$$\begin{aligned}
& d \in \mathbb{N} \\
& 0 < d \\
& \text{COLOR} = \{\text{green}, \text{red}\} \\
& \text{green} \neq \text{red} \\
& n \in \mathbb{N} \\
& n \leq d \\
& a \in \mathbb{N} \\
& b \in \mathbb{N} \\
& c \in \mathbb{N} \\
& a + b + c = n \\
& a = 0 \vee c = 0 \\
& ml_tl \in \text{COLOR} \\
& il_tl \in \text{COLOR} \\
& ml_tl = \text{green} \Rightarrow a + b < d \wedge c = 0 \\
& il_tl = \text{green} \Rightarrow 0 < b \wedge a = 0 \\
& il_tl = \text{green} \\
\vdash & ml_tl = \text{green} \Rightarrow a + b - 1 < d \wedge c + 1 = 0
\end{aligned}$$

MON

$$\begin{aligned}
& \text{green} \neq \text{red} \\
& ml_tl = \text{green} \Rightarrow a + b < d \wedge c = 0 \\
& il_tl = \text{green} \\
\vdash & ml_tl = \text{green} \Rightarrow a + b - 1 < d \wedge \\
& \quad c + 1 = 0
\end{aligned}$$

IMP_R . .

$d \in \mathbb{N}$
 $0 < d$
 $COLOR = \{\text{green}, \text{red}\}$
 $\text{green} \neq \text{red}$
 $n \in \mathbb{N}$
 $n \leq d$
 $a \in \mathbb{N}$
 $b \in \mathbb{N}$
 $c \in \mathbb{N}$
 $a + b + c = n$
 $a = 0 \vee c = 0$
 $ml_tl \in COLOR$
 $il_tl \in COLOR$
 $ml_tl = \text{green} \Rightarrow a + b < d \wedge c = 0$
 $il_tl = \text{green} \Rightarrow 0 < b \wedge a = 0$
 $il_tl = \text{green}$
 \vdash
 $ml_tl = \text{green} \Rightarrow a + b - 1 < d \wedge c + 1 = 0$

MON

$\text{green} \neq \text{red}$
 $ml_tl = \text{green} \Rightarrow a + b < d \wedge c = 0$
 $il_tl = \text{green}$
 \vdash
 $ml_tl = \text{green} \Rightarrow a + b - 1 < d \wedge$
 $c + 1 = 0$

IMP_R . . .

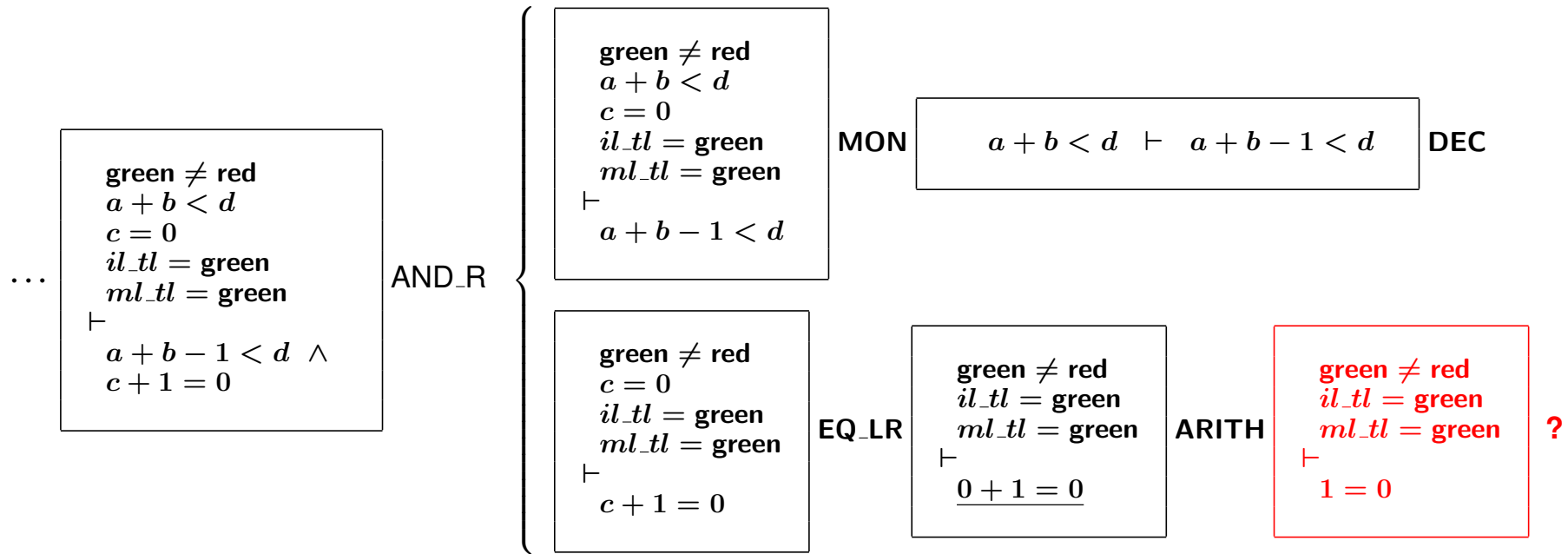
$\text{green} \neq \text{red}$
 $ml_tl = \text{green} \Rightarrow a + b < d \wedge c = 0$
 $il_tl = \text{green}$
 $ml_tl = \text{green}$
 \vdash
 $a + b - 1 < d \wedge c + 1 = 0$

...

IMP_L

$\text{green} \neq \text{red}$
 $a + b < d \wedge c = 0$
 $il_tl = \text{green}$
 $ml_tl = \text{green}$
 \vdash
 $a + b - 1 < d \wedge$
 $c + 1 = 0$

AND_L ...



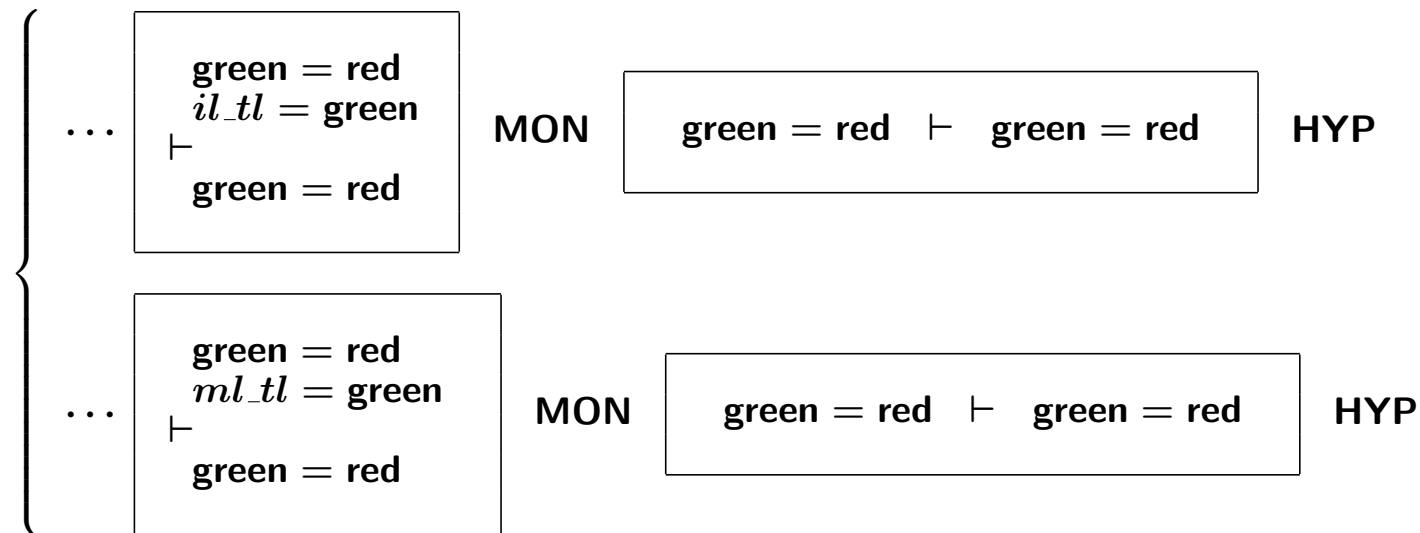
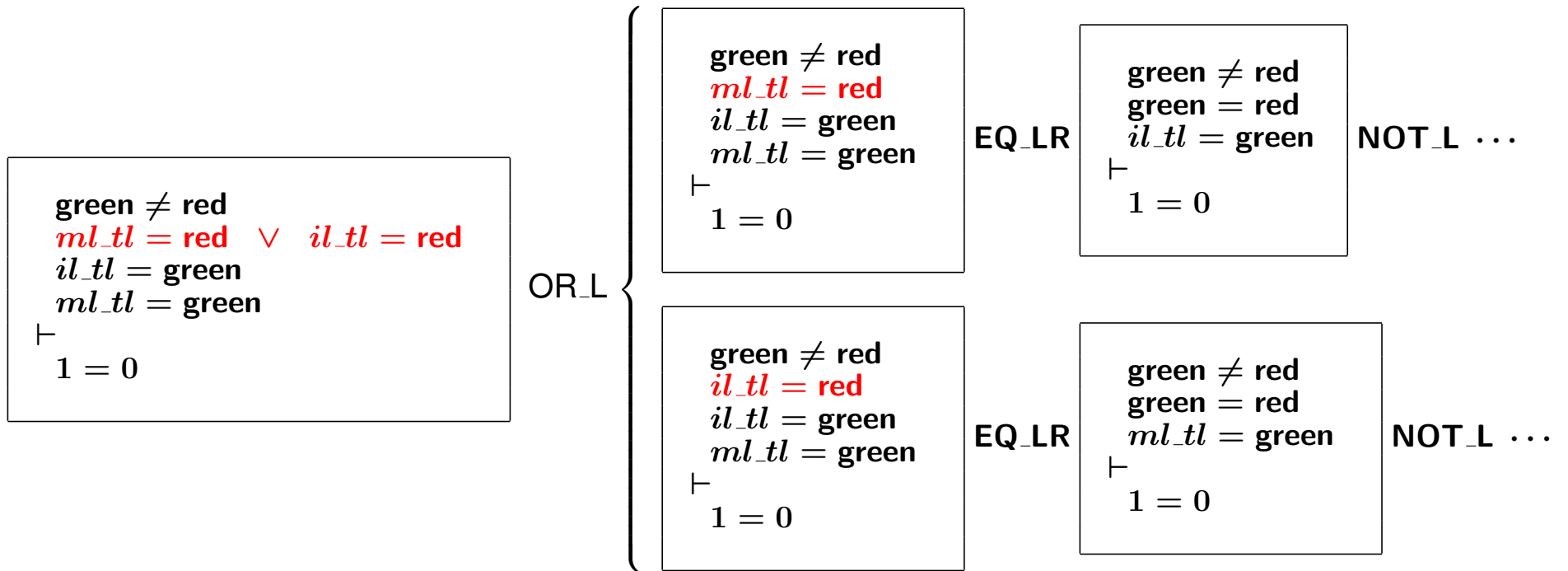
- In both cases, we were stopped by attempting to prove the following

$$\begin{array}{l} \text{green} \neq \text{red} \\ il_tl = \text{green} \\ ml_tl = \text{green} \\ \vdash \\ 1 = 0 \end{array}$$

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

$$\text{inv2_5: } ml_tl = \text{red} \vee il_tl = \text{red}$$



inv2_5: $ml_tl = red \vee il_tl = red$

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

- 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

axm0_1
 axm0_2
 axm2_1
 axm2_2
 inv0_1
 inv0_2
 inv1_1
 inv1_2
 inv1_3
 inv1_4
 inv1_5
 inv2_1
 inv2_2
 inv2_3
 inv2_4
 Guard of ML_out
 \vdash
 Modified inv2_3

$d \in \mathbb{N}$
 $0 < d$
 $COLOR = \{\text{green}, \text{red}\}$
 $\text{green} \neq \text{red}$
 $n \in \mathbb{N}$
 $n \leq d$
 $a \in \mathbb{N}$
 $b \in \mathbb{N}$
 $c \in \mathbb{N}$
 $a + b + c = n$
 $a = 0 \vee c = 0$
 $ml_tl \in COLOR$
 $il_tl \in COLOR$
 $ml_tl = \text{green} \Rightarrow a + b < d \wedge c = 0$
 $il_tl = \text{green} \Rightarrow 0 < b \wedge a = 0$
 $ml_tl = \text{green}$
 \vdash
 $ml_tl = \text{green} \Rightarrow a + 1 + b < d \wedge c = 0$

ML_out / inv2_3 / INV

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

$$\begin{array}{l}
 d \in \mathbb{N} \\
 0 < d \\
 \mathit{COLOR} = \{\mathit{green}, \mathit{red}\} \\
 \mathit{green} \neq \mathit{red} \\
 n \in \mathbb{N} \\
 n \leq d \\
 a \in \mathbb{N} \\
 b \in \mathbb{N} \\
 c \in \mathbb{N} \\
 a + b + c = n \\
 a = 0 \vee c = 0 \\
 \mathit{ml_tl} \in \mathit{COLOR} \\
 \mathit{il_tl} \in \mathit{COLOR} \\
 \mathit{ml_tl} = \mathit{green} \Rightarrow a + b < d \wedge c = 0 \\
 \mathit{il_tl} = \mathit{green} \Rightarrow 0 < b \wedge a = 0 \\
 \mathit{ml_tl} = \mathit{green} \\
 \vdash \\
 \mathit{ml_tl} = \mathit{green} \Rightarrow a + 1 + b < d \wedge \\
 \qquad \qquad \qquad c = 0
 \end{array}$$

MON

$$\begin{array}{l}
 \mathit{ml_tl} = \mathit{green} \Rightarrow a + b < d \wedge c = 0 \\
 \vdash \\
 \mathit{ml_tl} = \mathit{green} \Rightarrow a + 1 + b < d \wedge c = 0
 \end{array}$$

IMP_R...

$d \in \mathbb{N}$
 $0 < d$
 $COLOR = \{\text{green}, \text{red}\}$
 $\text{green} \neq \text{red}$
 $n \in \mathbb{N}$
 $n \leq d$
 $a \in \mathbb{N}$
 $b \in \mathbb{N}$
 $c \in \mathbb{N}$
 $a + b + c = n$
 $a = 0 \vee c = 0$
 $ml_tl \in COLOR$
 $il_tl \in COLOR$
 $ml_tl = \text{green} \Rightarrow a + b < d \wedge c = 0$
 $il_tl = \text{green} \Rightarrow 0 < b \wedge a = 0$
 $ml_tl = \text{green}$
 \vdash
 $ml_tl = \text{green} \Rightarrow a + 1 + b < d \wedge c = 0$

MON

$ml_tl = \text{green} \Rightarrow a + b < d \wedge c = 0$
 \vdash
 $ml_tl = \text{green} \Rightarrow a + 1 + b < d \wedge c = 0$

IMP_R...

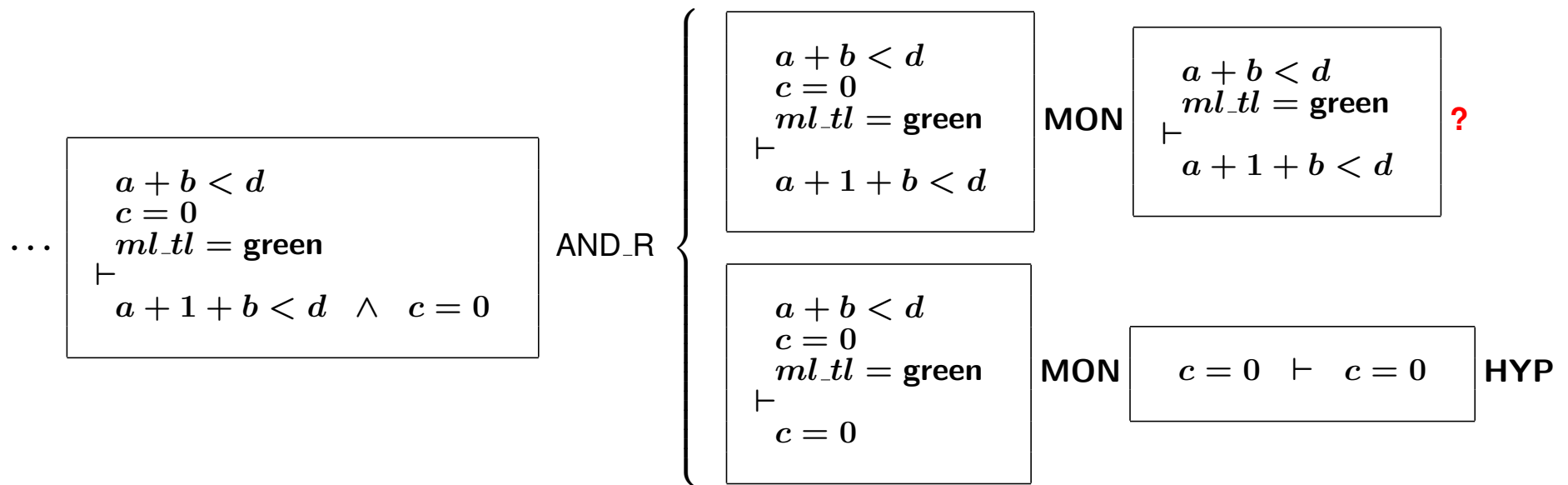
$ml_tl = \text{green} \Rightarrow a + b < d \wedge c = 0$
 $ml_tl = \text{green}$
 \vdash
 $a + 1 + b < d \wedge c = 0$

...

IMP_L

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

AND_L ...



- 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
    
```

```
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
```

- 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)

axm0_1
 axm0_2
 axm2_1
 axm2_2
 inv0_1
 inv0_2
 inv1_1
 inv1_2
 inv1_3
 inv1_4
 inv1_5
 inv2_1
 inv2_2
 inv2_3
 inv2_4
 Guard of ML_out_1

⊢
 Modified inv2_3

$d \in \mathbb{N}$
 $0 < d$
 $COLOR = \{\text{green}, \text{red}\}$
 $\text{green} \neq \text{red}$
 $n \in \mathbb{N}$
 $n \leq d$
 $a \in \mathbb{N}$
 $b \in \mathbb{N}$
 $c \in \mathbb{N}$
 $a + b + c = n$
 $a = 0 \vee c = 0$
 $ml_tl \in COLOR$
 $il_tl \in COLOR$
 $ml_tl = \text{green} \Rightarrow a + b < d \wedge c = 0$
 $il_tl = \text{green} \Rightarrow 0 < b \wedge a = 0$
 $ml_tl = \text{green}$
 $a + 1 + b < d$

⊢
 $ml_tl = \text{green} \Rightarrow a + 1 + b < d \wedge c = 0$

ML_out_1 / inv2_3 / INV

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

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

MON

$$\begin{array}{l}
ml_tl = \text{green} \Rightarrow a + b < d \wedge c = 0 \\
a + 1 + b < d \\
\vdash \\
ml_tl = \text{green} \Rightarrow a + 1 + b < d \wedge c = 0
\end{array}$$

IMP_R...

$d \in \mathbb{N}$
 $0 < d$
 $COLOR = \{\text{green}, \text{red}\}$
 $\text{green} \neq \text{red}$
 $n \in \mathbb{N}$
 $n \leq d$
 $a \in \mathbb{N}$
 $b \in \mathbb{N}$
 $c \in \mathbb{N}$
 $a + b + c = n$
 $a = 0 \vee c = 0$
 $ml_tl \in COLOR$
 $il_tl \in COLOR$
 $ml_tl = \text{green} \Rightarrow a + b < d \wedge c = 0$
 $il_tl = \text{green} \Rightarrow 0 < b \wedge a = 0$
 $ml_tl = \text{green}$
 $a + 1 + b < d$
 \vdash
 $ml_tl = \text{green} \Rightarrow a + 1 + b < d \wedge c = 0$

MON

$ml_tl = \text{green} \Rightarrow a + b < d \wedge c = 0$
 $a + 1 + b < d$
 \vdash
 $ml_tl = \text{green} \Rightarrow a + 1 + b < d \wedge c = 0$

IMP_R...

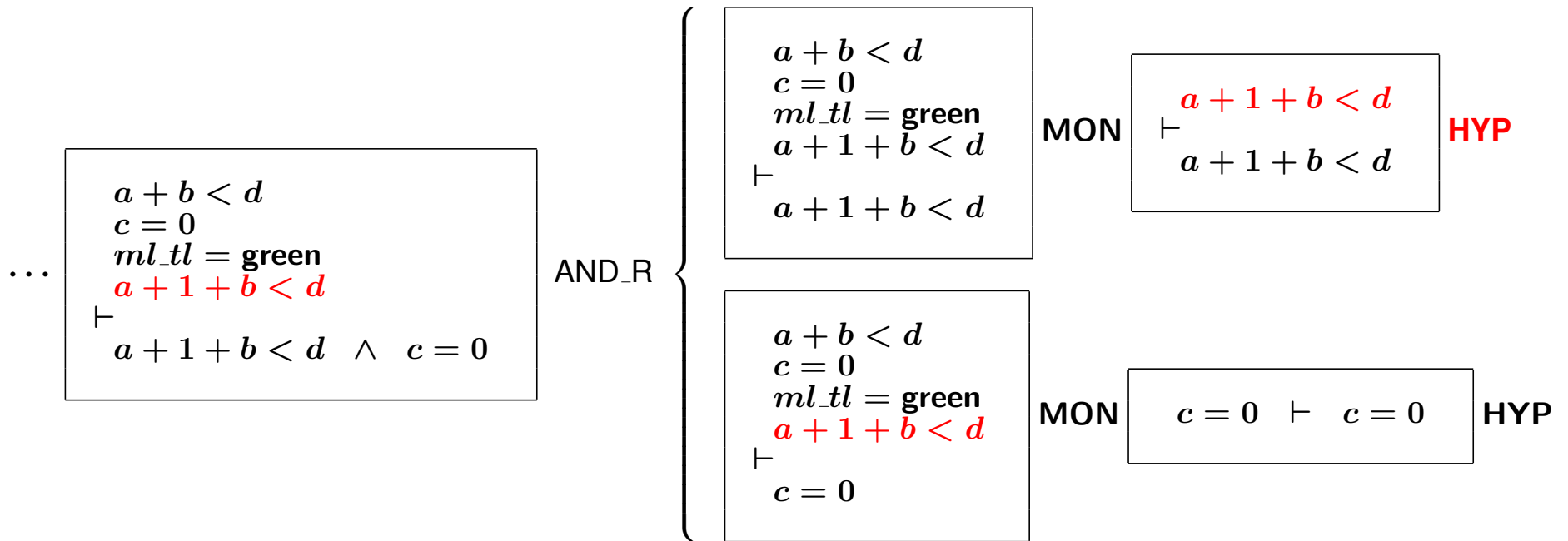
$ml_tl = \text{green} \Rightarrow a + b < d \wedge c = 0$
 $ml_tl = \text{green}$
 $a + 1 + b < d$
 \vdash
 $a + 1 + b < d \wedge c = 0$

...

IMP_L

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

AND_L ...



axm0_1
 axm0_2
 axm2_1
 axm2_2
 inv0_1
 inv0_2
 inv1_1
 inv1_2
 inv1_3
 inv1_4
 inv1_5
 inv2_1
 inv2_2
 inv2_3
 inv2_4
 Guard of ML_out_2

⊢
 Modified **inv2_3**

$d \in \mathbb{N}$
 $0 < d$
 $COLOR = \{\text{green}, \text{red}\}$
 $\text{green} \neq \text{red}$
 $n \in \mathbb{N}$
 $n \leq d$
 $a \in \mathbb{N}$
 $b \in \mathbb{N}$
 $c \in \mathbb{N}$
 $a + b + c = n$
 $a = 0 \vee c = 0$
 $ml_tl \in COLOR$
 $il_tl \in COLOR$
 $ml_tl = \text{green} \Rightarrow a + b < d \wedge c = 0$
 $il_tl = \text{green} \Rightarrow 0 < b \wedge a = 0$
 $ml_tl = \text{green}$
 $a + 1 + b = d$
 ⊢
 $\text{red} = \text{green} \Rightarrow a + 1 + b < d \wedge c = 0$

ML_out_2 / inv2_3 / INV

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

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

MON

$$\begin{array}{l}
\text{green} \neq \text{red} \\
\vdash \\
\text{red} = \text{green} \Rightarrow a + 1 + b < d \wedge c = 0
\end{array}$$

IMP_R

$d \in \mathbb{N}$
 $0 < d$
 $COLOR = \{\text{green}, \text{red}\}$
 $\text{green} \neq \text{red}$
 $n \in \mathbb{N}$
 $n \leq d$
 $a \in \mathbb{N}$
 $b \in \mathbb{N}$
 $c \in \mathbb{N}$
 $a + b + c = n$
 $a = 0 \vee c = 0$
 $ml_tl \in COLOR$
 $il_tl \in COLOR$
 $ml_tl = \text{green} \Rightarrow a + b < d \wedge c = 0$
 $il_tl = \text{green} \Rightarrow 0 < b \wedge a = 0$
 $ml_tl = \text{green}$
 $a + 1 + b = d$
 \vdash
 $\text{red} = \text{green} \Rightarrow a + 1 + b < d \wedge c = 0$

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

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

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

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

- 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
 axm0_2
 axm2_1
 axm2_2
 inv0_1
 inv0_2
 inv1_1
 inv1_2
 inv1_3
 inv1_4
 inv1_5
 inv2_1
 inv2_2
 inv2_3
 inv2_4
 Guard of event IL_out
 \vdash
 Modified invariant **inv2_4**

$$\begin{aligned}
 & d \in \mathbb{N} \\
 & 0 < d \\
 & COLOR = \{\text{green}, \text{red}\} \\
 & \text{green} \neq \text{red} \\
 & n \in \mathbb{N} \\
 & n \leq d \\
 & a \in \mathbb{N} \\
 & b \in \mathbb{N} \\
 & c \in \mathbb{N} \\
 & a + b + c = n \\
 & a = 0 \vee c = 0 \\
 & ml_tl \in COLOR \\
 & il_tl \in COLOR \\
 & ml_tl = \text{green} \Rightarrow a + b < d \wedge c = 0 \\
 & il_tl = \text{green} \Rightarrow 0 < b \wedge a = 0 \\
 & il_tl = \text{green} \\
 \vdash & \\
 & il_tl = \text{green} \Rightarrow 0 < b - 1 \wedge a = 0
 \end{aligned}$$

IL_out / inv2_4 / INV

```

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

$d \in \mathbb{N}$
 $0 < d$
 $COLOR = \{\text{green}, \text{red}\}$
 $\text{green} \neq \text{red}$
 $n \in \mathbb{N}$
 $n \leq d$
 $a \in \mathbb{N}$
 $b \in \mathbb{N}$
 $c \in \mathbb{N}$
 $a + b + c = n$
 $a = 0 \vee c = 0$
 $ml_tl \in COLOR$
 $il_tl \in COLOR$
 $ml_tl = \text{green} \Rightarrow a + b < d \wedge c = 0$
 $il_tl = \text{green} \Rightarrow 0 < b \wedge a = 0$
 $il_tl = \text{green}$
 \vdash
 $il_tl = \text{green} \Rightarrow 0 < b - 1 \wedge a = 0$

MON

$il_tl = \text{green} \Rightarrow 0 < b \wedge a = 0$
 $il_tl = \text{green}$
 \vdash
 $il_tl = \text{green} \Rightarrow 0 < b - 1 \wedge a = 0$

IMP_R

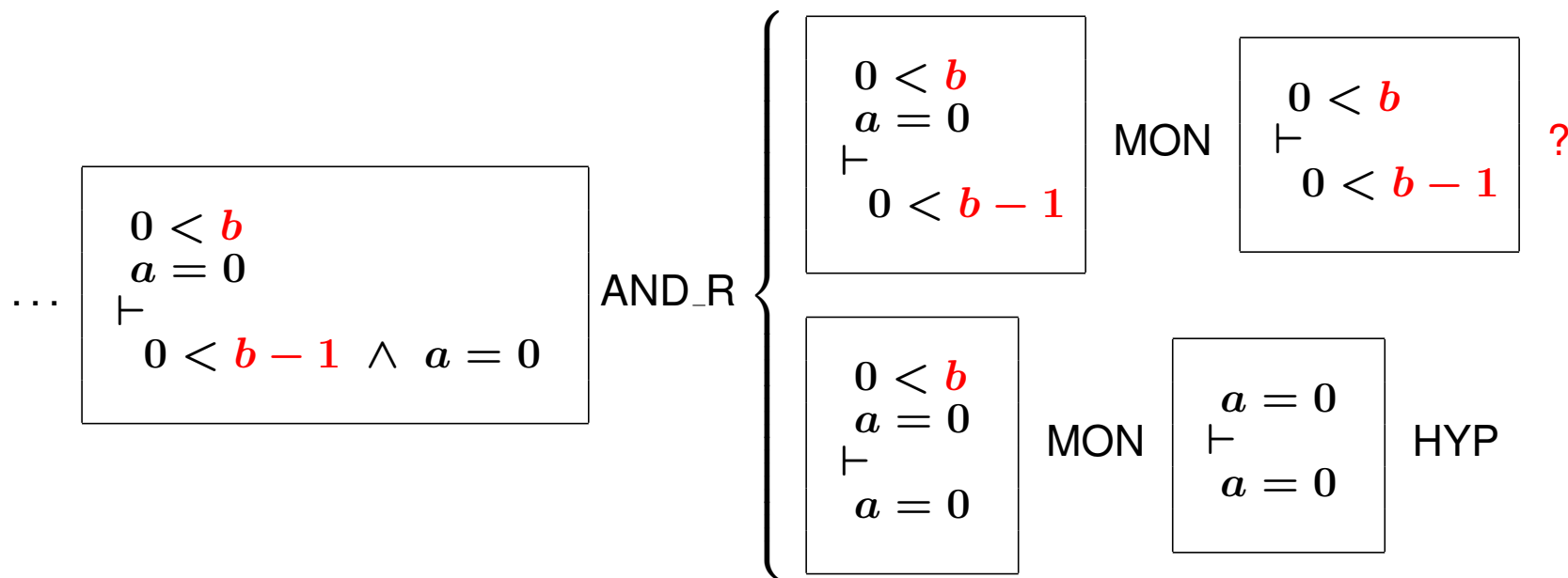
...

$il_tl = \text{green} \Rightarrow 0 < b \wedge a = 0$
 $il_tl = \text{green}$
 \vdash
 $0 < b - 1 \wedge a = 0$

IMP_L

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

AND_L



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

```

IL_out_1
when
  when
    il_tl = green
    b ≠ 1
  then
    b, c := b - 1, c + 1
  end
end

```

```

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

```

```
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
 axm0_2
 axm2_1
 axm2_2
 inv0_1
 inv0_2
 inv1_1
 inv1_2
 inv1_3
 inv1_4
 inv1_5
 inv2_1
 inv2_2
 inv2_3
 inv2_4

Guard of event IL_out_1

\vdash
 Modified invariant **inv2_4**

$d \in \mathbb{N}$
 $0 < d$
 $COLOR = \{\text{green}, \text{red}\}$
 $\text{green} \neq \text{red}$
 $n \in \mathbb{N}$
 $n \leq d$
 $a \in \mathbb{N}$
 $b \in \mathbb{N}$
 $c \in \mathbb{N}$
 $a + b + c = n$
 $a = 0 \vee c = 0$
 $ml_tl \in COLOR$
 $il_tl \in COLOR$
 $ml_tl = \text{green} \Rightarrow a + b < d \wedge c = 0$
 $il_tl = \text{green} \Rightarrow 0 < b \wedge a = 0$
 $il_tl = \text{green}$
 $b \neq 1$
 \vdash
 $il_tl = \text{green} \Rightarrow 0 < b - 1 \wedge a = 0$

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

$d \in \mathbb{N}$
 $0 < d$
 $COLOR = \{\text{green}, \text{red}\}$
 $\text{green} \neq \text{red}$
 $n \in \mathbb{N}$
 $n \leq d$
 $a \in \mathbb{N}$
 $b \in \mathbb{N}$
 $c \in \mathbb{N}$
 $a + b + c = n$
 $a = 0 \vee c = 0$
 $ml_tl \in COLOR$
 $il_tl \in COLOR$
 $ml_tl = \text{green} \Rightarrow a + b < d \wedge c = 0$
 $il_tl = \text{green} \Rightarrow 0 < b \wedge a = 0$
 $il_tl = \text{green}$
 $b \neq 1$
 \vdash
 $il_tl = \text{green} \Rightarrow 0 < b - 1 \wedge a = 0$

MON

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

IMP_R

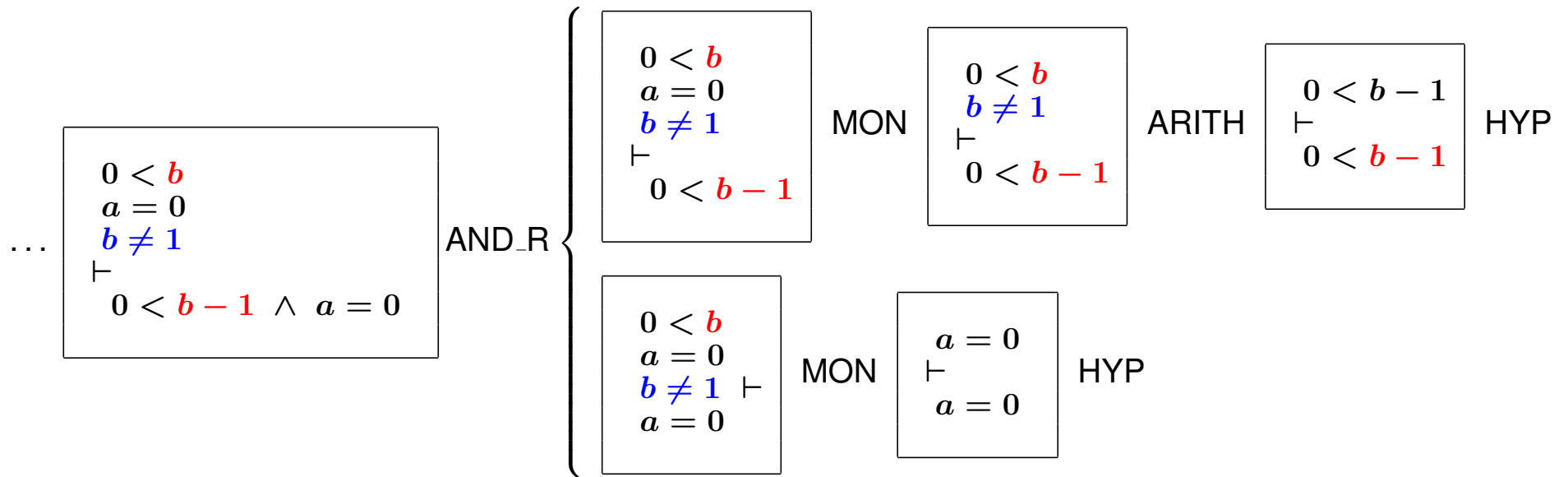
...

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

IMP_L

$0 < b \wedge a = 0$
 $b \neq 1$
 \vdash
 $0 < b - 1 \wedge a = 0$

AND_L



axm0_1
 axm0_2
 axm2_1
 axm2_2
 inv0_1
 inv0_2
 inv1_1
 inv1_2
 inv1_3
 inv1_4
 inv1_5
 inv2_1
 inv2_2
 inv2_3
 inv2_4
 Guard of event IL_out_2

\vdash
 Modified invariant **inv2_4**

$d \in \mathbb{N}$
 $0 < d$
 $COLOR = \{\text{green}, \text{red}\}$
 $\text{green} \neq \text{red}$
 $n \in \mathbb{N}$
 $n \leq d$
 $a \in \mathbb{N}$
 $b \in \mathbb{N}$
 $c \in \mathbb{N}$
 $a + b + c = n$
 $a = 0 \vee c = 0$
 $ml_tl \in COLOR$
 $il_tl \in COLOR$
 $ml_tl = \text{green} \Rightarrow a + b < d \wedge c = 0$
 $il_tl = \text{green} \Rightarrow 0 < b \wedge a = 0$
 $il_tl = \text{green}$
 $b = 1$
 \vdash
 $\text{red} = \text{green} \Rightarrow 0 < b - 1 \wedge a = 0$

IL_out_1 / **inv2_4** / INV

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

$d \in \mathbb{N}$
 $0 < d$
 $COLOR = \{\text{green}, \text{red}\}$
 $\text{green} \neq \text{red}$
 $n \in \mathbb{N}$
 $n \leq d$
 $a \in \mathbb{N}$
 $b \in \mathbb{N}$
 $c \in \mathbb{N}$
 $a + b + c = n$
 $a = 0 \vee c = 0$
 $ml_tl \in COLOR$
 $il_tl \in COLOR$
 $ml_tl = \text{green} \Rightarrow a + b < d \wedge c = 0$
 $il_tl = \text{green} \Rightarrow 0 < b \wedge a = 0$
 $il_tl = \text{green}$
 $b = 1$
 \vdash
 $\text{red} = \text{green} \Rightarrow 0 < b - 1 \wedge a = 0$

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

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

EQ_LR

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

NOT_L

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

EQL

-
- 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

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

$$\text{inv2_5: } ml_tl = \text{red} \vee il_tl = \text{red}$$

Unless we correct them as follows:

```
ML_tl_green
when
  ml_tl = red
  a + b < d
  c = 0
then
  ml_tl := green
  il_tl := red
end
```

```
IL_tl_green
when
  il_tl = red
  0 < b
  a = 0
then
  il_tl := green
  ml_tl := red
end
```

- Correct event refinement: **OK**
- Absence of divergence of new events: **FAILURE**
- Absence of deadlock: **?**

ML_tl_green

when

$ml_tl = red$

$a + b < d$

$c = 0$

then

$ml_tl := green$

$il_tl := red$

end

IL_tl_green

when

$il_tl = red$

$0 < b$

$a = 0$

then

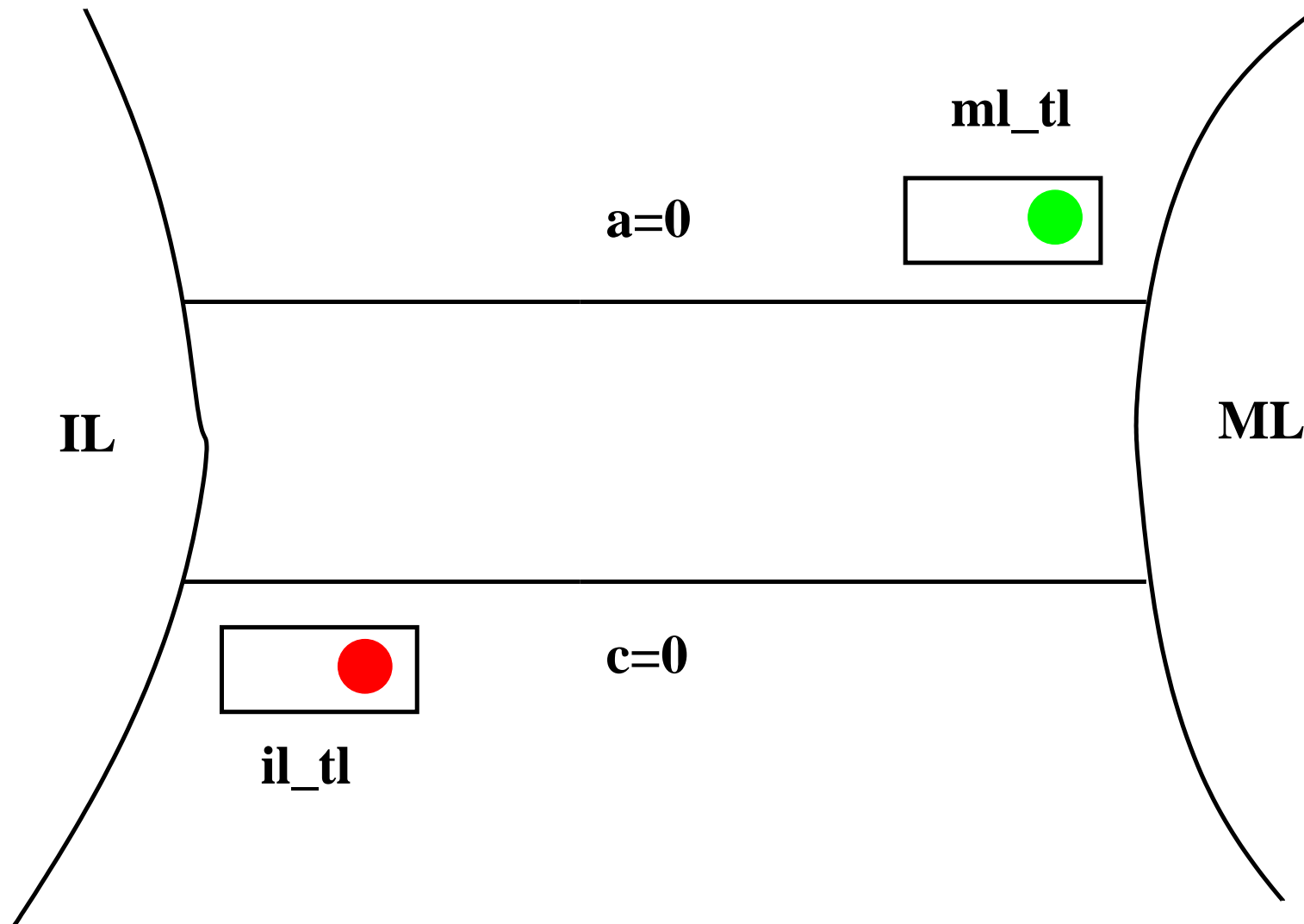
$il_tl := green$

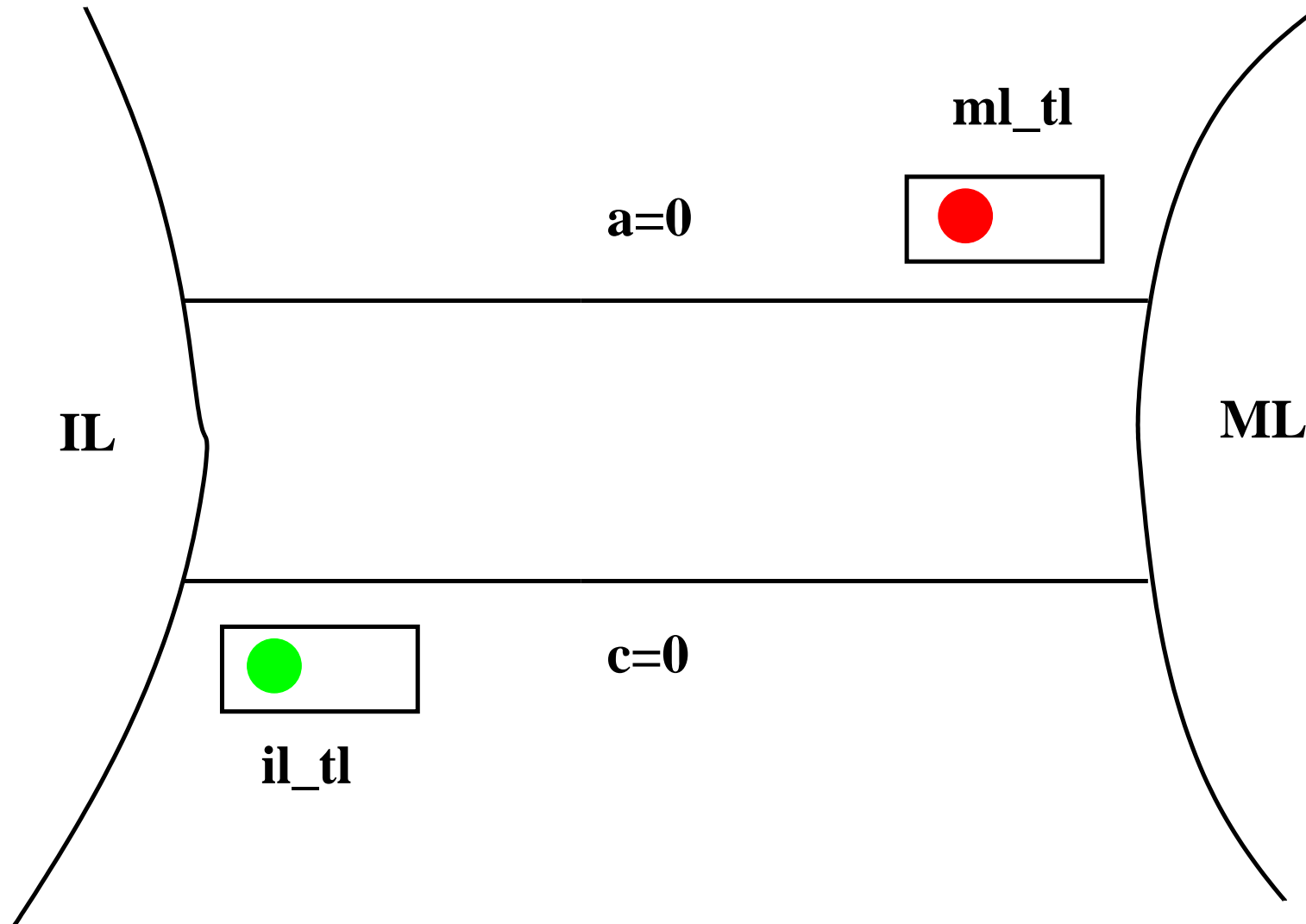
$ml_tl := red$

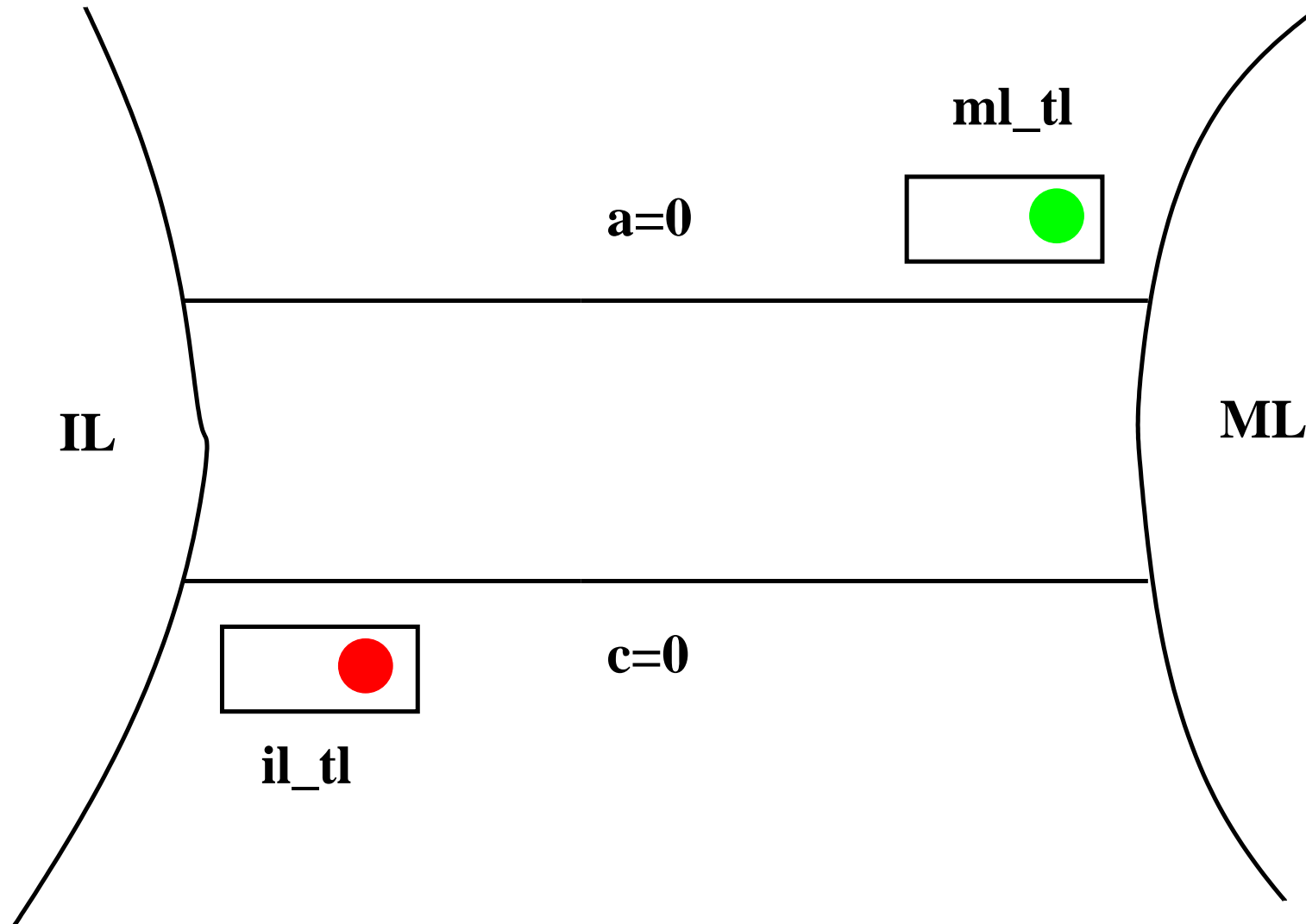
end

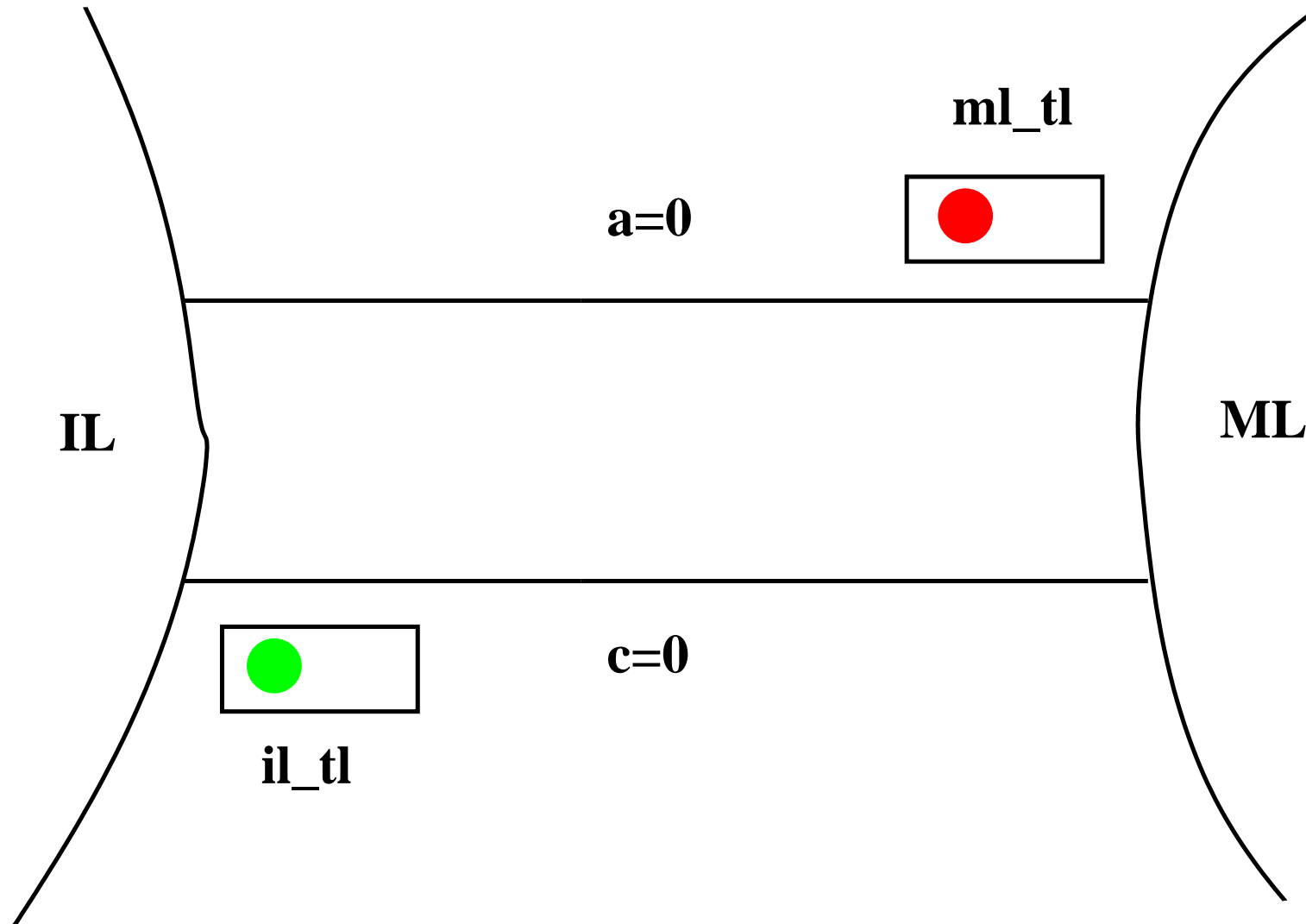
When a and c are both equal to 0 and b is positive, then both events are always alternatively enabled

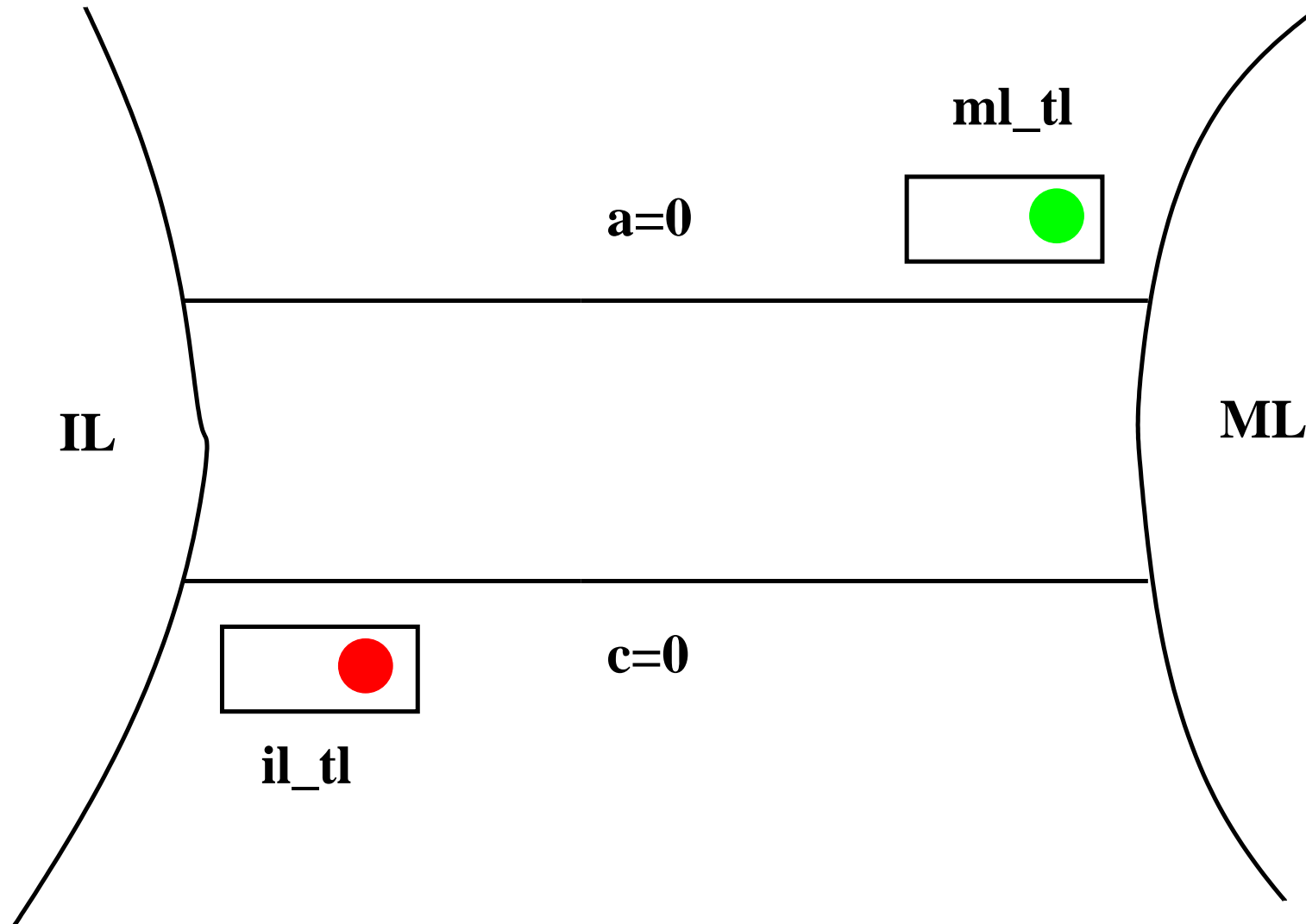
The lights can change colors very rapidly

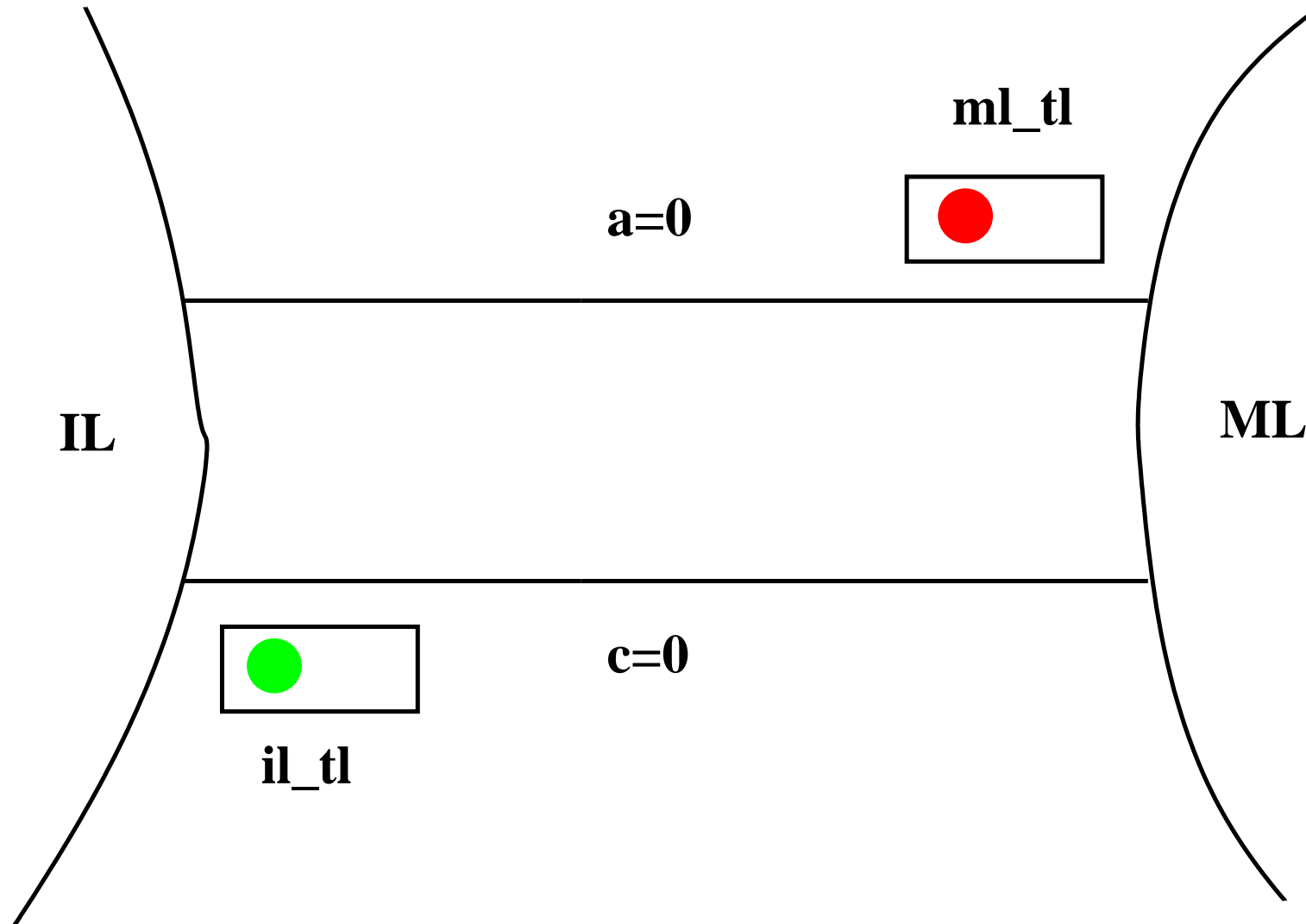












-
- Allowing each light **to turn green** only when at least one car has **passed in the other direction**
 - For this, we introduce **two additional variables**:

inv2_6: $ml_pass \in \{0, 1\}$

inv2_7: $il_pass \in \{0, 1\}$

ML_out_1

when

$ml_tl = \text{green}$

$a + 1 + b < d$

then

$a := a + 1$

$ml_pass := 1$

end

ML_out_2

when

$ml_tl = \text{green}$

$a + 1 + b = d$

then

$a := a + 1$

$ml_tl := \text{red}$

$ml_pass := 1$

end

```
IL_out_1
  when
    il_tl = green
    b ≠ 1
  then
    b := b - 1
    c := c + 1
    il_pass := 1
  end
```

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

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

```
IL_tl_green
when
  il_tl = red
   $0 < b$ 
   $a = 0$ 
  ml_pass = 1
then
  il_tl := green
  ml_tl := red
  il_pass := 0
end
```

We exhibit the following variant

variant_2: $ml_pass + il_pass$

$$ml_tl = red$$

$$a + b < d$$

$$c = 0$$

$$il_pass = 1$$

$$\Rightarrow$$

$$il_pass + 0 <$$

$$ml_pass + il_pass$$

$$il_tl = red$$

$$b > 0$$

$$a = 0$$

$$ml_pass = 1$$

$$\Rightarrow$$

$$ml_pass + 0 <$$

$$ml_pass + il_pass$$

This cannot be proved. This suggests the following invariants:

$$\mathbf{inv2_8:} \quad ml_tl = red \Rightarrow ml_pass = 1$$

$$\mathbf{inv2_9:} \quad il_tl = red \Rightarrow il_pass = 1$$

$$0 < d$$

$$ml_tl \in \{\text{red}, \text{green}\}$$

$$il_tl \in \{\text{red}, \text{green}\}$$

$$ml_pass \in \{0, 1\}$$

$$il_pass \in \{0, 1\}$$

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

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

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

$$ml_tl = \text{red} \Rightarrow ml_pass = 1$$

$$il_tl = \text{red} \Rightarrow il_pass = 1$$

\Rightarrow

$$(ml_tl = \text{red} \wedge a + b < d \wedge c = 0 \wedge il_pass = 1) \vee$$

$$(il_tl = \text{red} \wedge a = 0 \wedge b > 0 \wedge ml_pass = 1) \vee$$

$$ml_tl = \text{green} \vee il_tl = \text{green} \vee a > 0 \vee c > 0$$

The previous statement reduces to the following, which is true

$$0 < d$$

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

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

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

$$\Rightarrow$$

$$(a + b < d \wedge c = 0) \vee$$

$$(a = 0 \wedge b > 0) \vee$$

$$a > 0 \vee$$

$$c > 0$$

$$\rightsquigarrow$$

$$0 < d$$

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

$$\Rightarrow$$

$$b < d \vee b > 0$$

- Thanks to the **proofs**:
 - We discovered 4 errors
 - We introduced several additional invariants
 - We corrected 4 events
 - We introduced 2 more variables

Axioms Abstract invariants Concrete invariants Concrete guards ┆ Same actions on common variables	SIM
--	-----

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: $il_tl = 1 \Rightarrow 0 < b \wedge a = 0$

inv2_5: $ml_tl = \text{red} \vee il_tl = \text{red}$

inv2_6: $ml_pass \in \{0, 1\}$

inv2_7: $il_pass \in \{0, 1\}$

inv2_8: $ml_tl = \text{red} \Rightarrow ml_pass = 1$

inv2_9: $il_tl = \text{red} \Rightarrow il_pass = 1$

variant2: $ml_pass + il_pass$

ML_out_1

when

$ml_tl = \text{green}$

$a + 1 + b < d$

then

$a := a + 1$

$ml_pass := 1$

end

ML_out_2

when

$ml_tl = \text{green}$

$a + 1 + b = d$

then

$a := a + 1$

$ml_pass := 1$

$ml_tl := \text{red}$

end

```
IL_out_1
  when
    il_tl = green
    b ≠ 1
  then
    b := b - 1
    c := c + 1
    il_pass := 1
  end
```

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

ML_tl_green

when

$ml_tl = red$

$a + b < d$

$c = 0$

$il_pass = 1$

then

$ml_tl := green$

$il_tl := red$

$ml_pass := 0$

end

IL_tl_green

when

$il_tl = red$

$0 < b$

$a = 0$

$ml_pass = 1$

then

$il_tl := green$

$ml_tl := red$

$il_pass := 0$

end

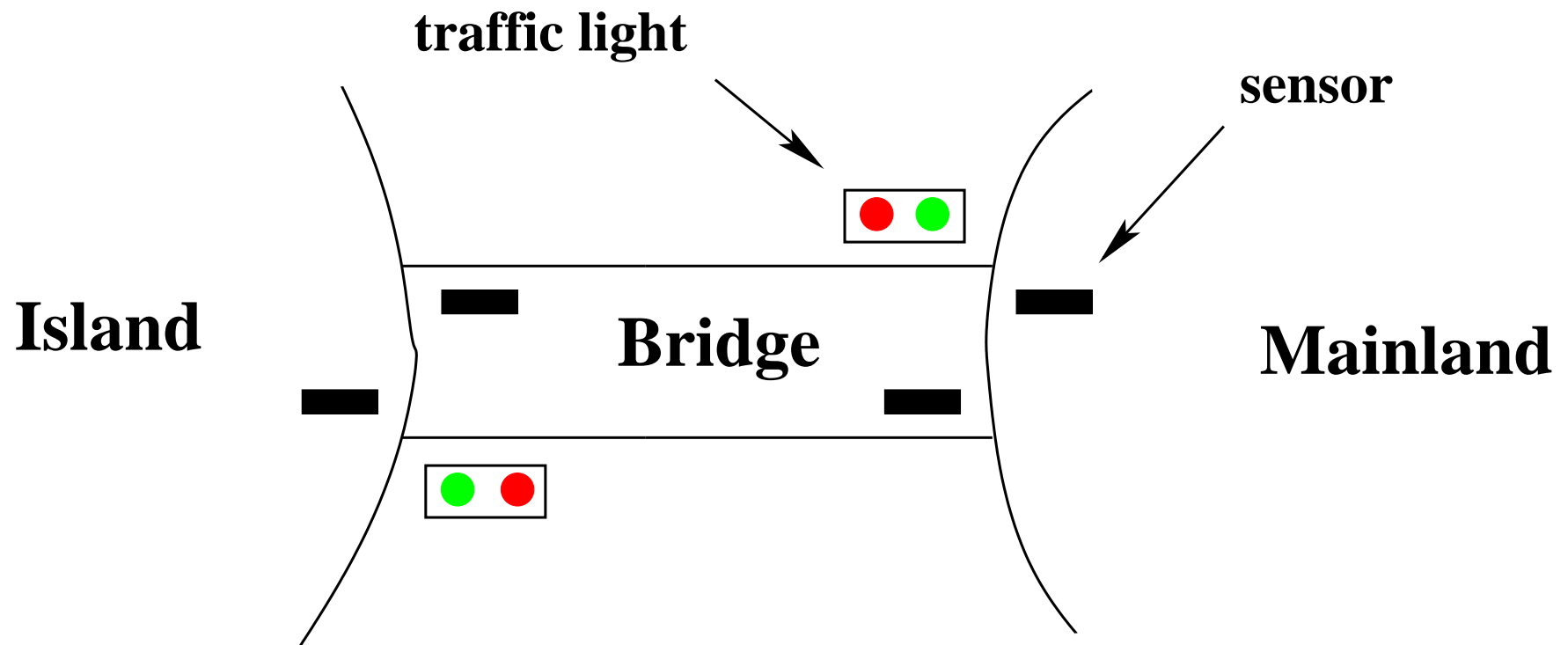
- These events are identical to their abstract versions

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

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

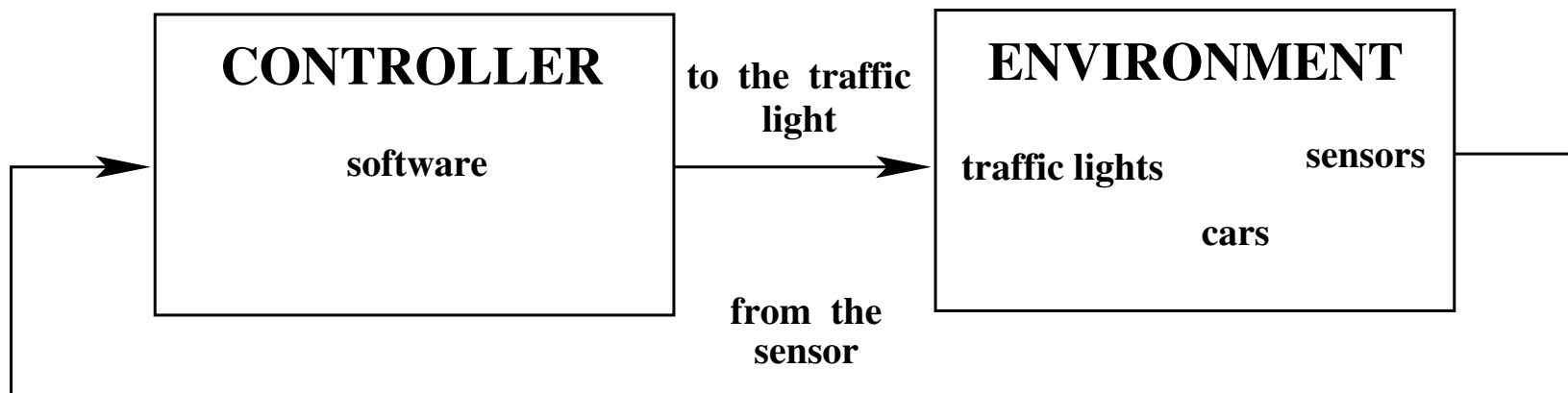
- **Initial model**: Limiting the number of cars (FUN_2)
- **First refinement**: Introducing the one way bridge (FUN_3)
- **Second refinement**: Introducing the traffic lights (EQP_1,2,3)
- **Third refinement**: Introducing the sensors (EQP_4,5)

Reminder of the **physical system**



-We want to **clearly identify** in our model:

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



Controller variables: *a*,
b,
c,
ml_pass,
il_pass

These **new variables** denote **physical objects**

Environment variables: *A*,

B,

C,

ML_OUT_SR,

ML_IN_SR,

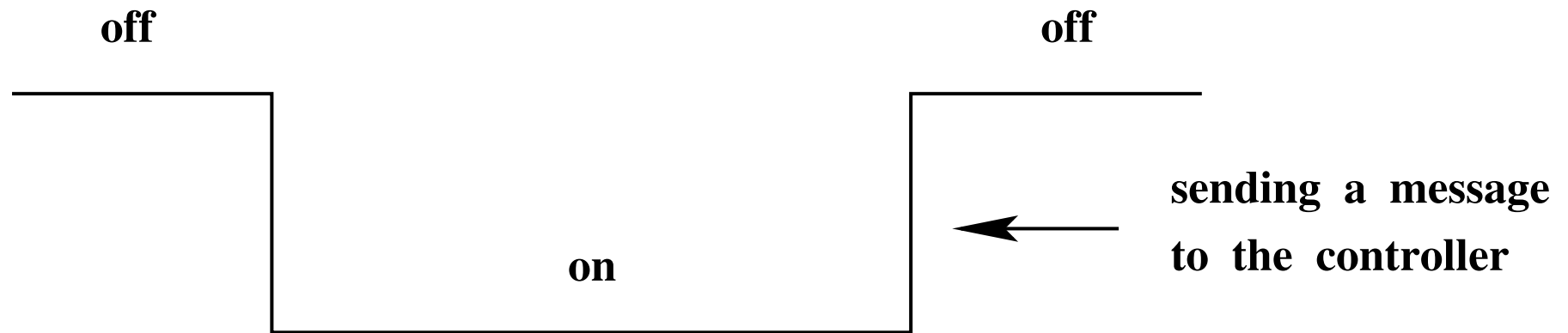
IL_OUT_SR,

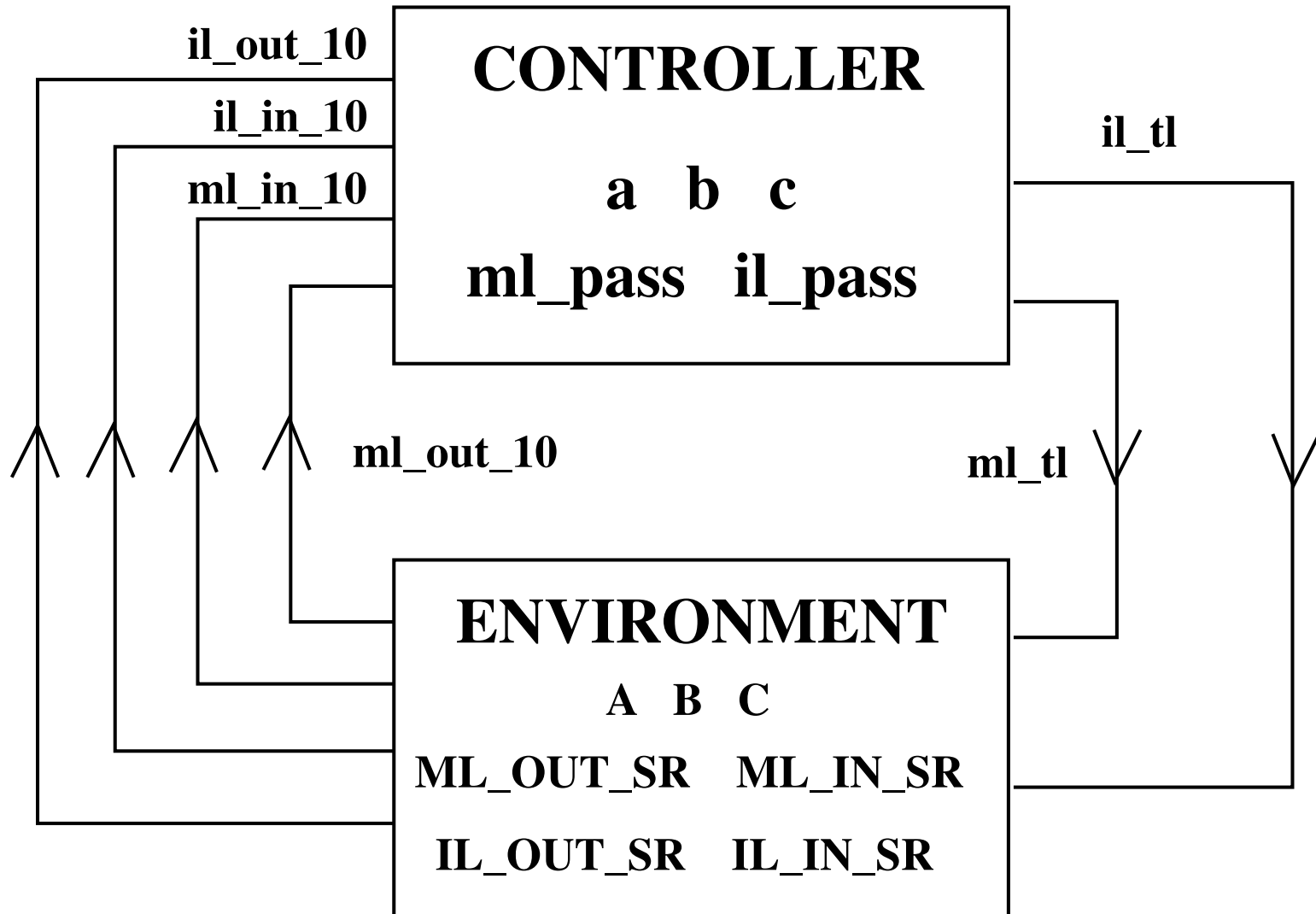
IL_IN_SR

Output channels: ml_tl ,
 il_tl

Input channels: *ml_out_10*,
ml_in_10,
il_in_10,
il_out_10

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





carrier sets: \dots, SENSOR

constants: $\dots, \mathit{on}, \mathit{off}$

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

axm3_2: $\mathit{on} \neq \mathit{off}$

inv3_1 : *ML_OUT_SR* ∈ *SENSOR*

inv3_2 : *ML_IN_SR* ∈ *SENSOR*

inv3_3 : *IL_OUT_SR* ∈ *SENSOR*

inv3_4 : *IL_IN_SR* ∈ *SENSOR*

inv3_5 : $A \in \mathbb{N}$

inv3_6 : $B \in \mathbb{N}$

inv3_7 : $C \in \mathbb{N}$

inv3_8 : $ml_out_10 \in \text{BOOL}$

inv3_9 : $ml_in_10 \in \text{BOOL}$

inv3_10 : $il_out_10 \in \text{BOOL}$

inv3_11 : $il_in_10 \in \text{BOOL}$

When sensors are on, there are cars on them

$$\text{inv3_12} : \text{IL_IN_SR} = \text{on} \Rightarrow A > 0$$

$$\text{inv3_13} : \text{IL_OUT_SR} = \text{on} \Rightarrow B > 0$$

$$\text{inv3_14} : \text{ML_IN_SR} = \text{on} \Rightarrow C > 0$$

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

EQP-5

Drivers obey the traffic lights

$$\text{inv3_15} : \text{ml_out_10} = \text{TRUE} \Rightarrow \text{ml_tl} = \text{green}$$
$$\text{inv3_16} : \text{il_out_10} = \text{TRUE} \Rightarrow \text{il_tl} = \text{green}$$

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

EQP-3

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

inv3_17 : $IL_IN_SR = on \Rightarrow il_in_10 = FALSE$

inv3_18 : $IL_OUT_SR = on \Rightarrow il_out_10 = FALSE$

inv3_19 : $ML_IN_SR = on \Rightarrow ml_in_10 = FALSE$

inv3_20 : $ML_OUT_SR = on \Rightarrow ml_out_10 = FALSE$

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

FUN-5

Linking the physical and logical cars (1)

$$\text{inv3_21 : } il_in_10 = \text{TRUE} \wedge ml_out_10 = \text{TRUE} \Rightarrow A = a$$

$$\text{inv3_22 : } il_in_10 = \text{FALSE} \wedge ml_out_10 = \text{TRUE} \Rightarrow A = a + 1$$

$$\text{inv3_23 : } il_in_10 = \text{TRUE} \wedge ml_out_10 = \text{FALSE} \Rightarrow A = a - 1$$

$$\text{inv3_24 : } il_in_10 = \text{FALSE} \wedge ml_out_10 = \text{FALSE} \Rightarrow A = a$$

Linking the physical and logical cars (2)

$$\text{inv3_25 : } il_in_10 = \text{TRUE} \wedge il_out_10 = \text{TRUE} \Rightarrow B = b$$

$$\text{inv3_26 : } il_in_10 = \text{TRUE} \wedge il_out_10 = \text{FALSE} \Rightarrow B = b + 1$$

$$\text{inv3_27 : } il_in_10 = \text{FALSE} \wedge il_out_10 = \text{TRUE} \Rightarrow B = b - 1$$

$$\text{inv3_28 : } il_in_10 = \text{FALSE} \wedge il_out_10 = \text{FALSE} \Rightarrow B = b$$

$$\text{inv3_29 : } il_out_10 = \text{TRUE} \wedge ml_out_10 = \text{TRUE} \Rightarrow C = c$$

$$\text{inv3_30 : } il_out_10 = \text{TRUE} \wedge ml_out_10 = \text{FALSE} \Rightarrow C = c + 1$$

$$\text{inv3_31 : } il_out_10 = \text{FALSE} \wedge ml_out_10 = \text{TRUE} \Rightarrow C = c - 1$$

$$\text{inv3_32 : } il_out_10 = \text{FALSE} \wedge ml_out_10 = \text{FALSE} \Rightarrow C = c$$

The basic properties hold for the physical cars

$$\text{inv3_33} : A = 0 \vee C = 0$$

$$\text{inv3_34} : A + B + C \leq d$$

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

```

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

```

```

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

```

```

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

```

```

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

```

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

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

```
(abstract-)IL_out_1
  when
     $il\_tl = \text{green}$ 
     $b \neq 1$ 
  then
     $b := b - 1$ 
     $c := c + 1$ 
     $il\_pass := 1$ 
  end
```

```
(abstract-)IL_out_2
  when
     $il\_tl = \text{green}$ 
     $b = 1$ 
  then
     $b := b - 1$ 
     $c := c + 1$ 
     $il\_pass := 1$ 
     $il\_tl := \text{red}$ 
  end
```

```

ML_in
  when
     $ml\_in\_10 = \text{TRUE}$ 
     $0 < c$ 
  then
     $c := c - 1$ 
     $ml\_in\_10 := \text{FALSE}$ 
  end

```

```

IL_in
  when
     $il\_in\_10 = \text{TRUE}$ 
     $0 < a$ 
  then
     $a := a - 1$ 
     $b := b + 1$ 
     $il\_in\_10 := \text{FALSE}$ 
  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

```

```

ML_tl_green
  when
    ml_tl = red
    a + b < d
    c = 0
    il_pass = 1
    il_out_10 = FALSE
  then
    ml_tl := green
    il_tl := red
    ml_pass := FALSE
  end

```

```

IL_tl_green
  when
    il_tl = red
    a = 0
    ml_pass = 1
    ml_out_10 = FALSE
  then
    il_tl := green
    ml_tl := red
    il_pass := 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

```

```
ML_out_arr
when
  ML_OUT_SR = off
  ml_out_10 = FALSE
then
  ML_OUT_SR := on
end
```

```
ML_in_arr
when
  ML_IN_SR = off
  ml_in_10 = FALSE
  C > 0
then
  ML_IN_SR := on
end
```

```
IL_in_arr
when
  IL_IN_SR = off
  il_in_10 = FALSE
  A > 0
then
  IL_IN_SR := on
end
```

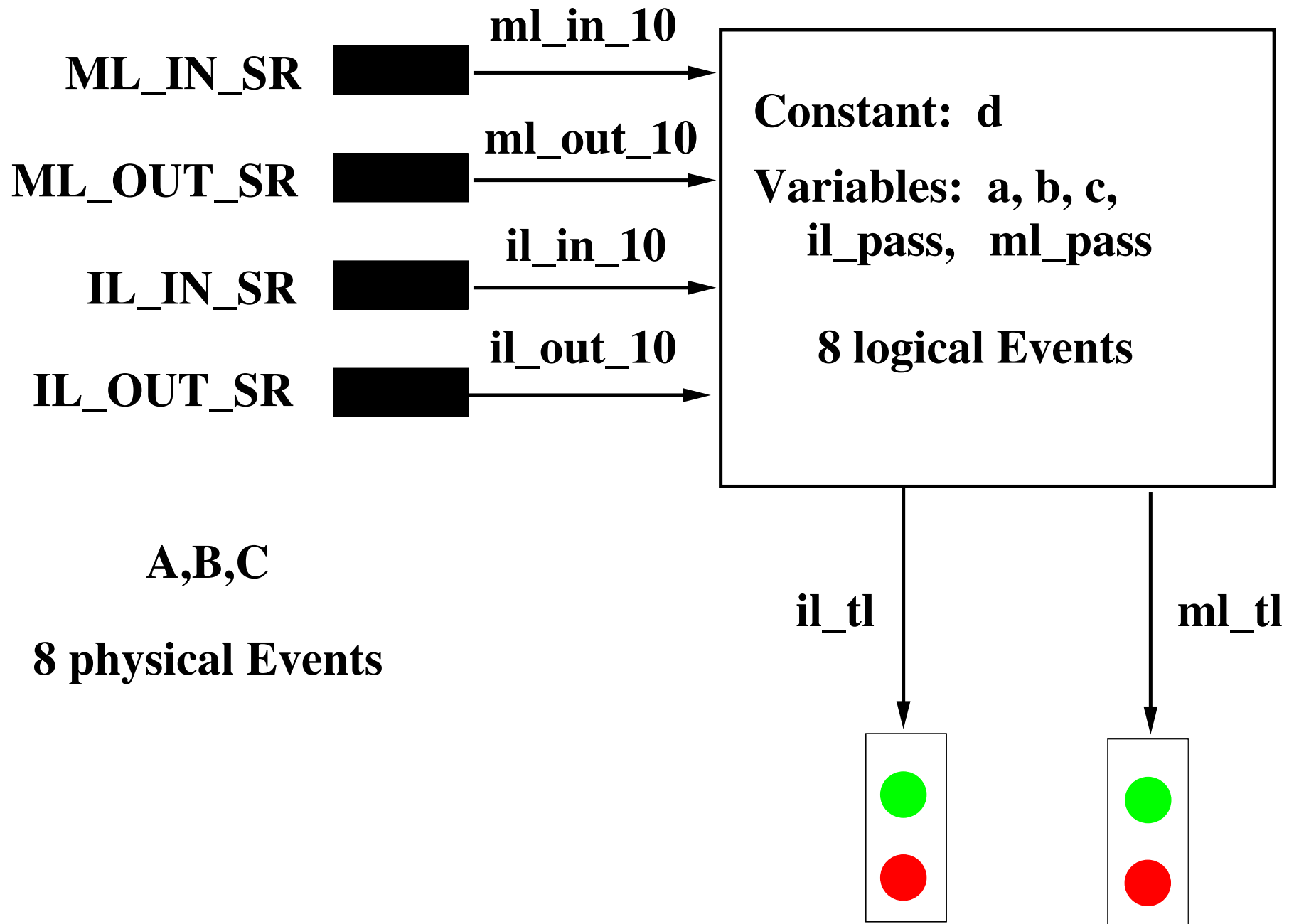
```
IL_out_arr
when
  IL_OUT_SR = off
  il_out_10 = FALSE
  B > 0
then
  IL_OUT_SR := on
end
```

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

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

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

```
IL_out_dep
  when
    IL_OUT_SR = on
    il_tl = green
  then
    IL_OUT_SR := off
    il_out_10 := TRUE
    B = B - 1
    C = C + 1
  end
```



- **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?

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

- **Who** is going to perform these proofs?
 - An automatic tool: **the Prover**
 - Sometimes helped by the Engineer (**interactive proving**)

-
- **Three basic tools:**
 - Proof Obligation Generator
 - Prover
 - Model translators into Hardware or Software languages
 - These tools are embedded into a **Development Data Base**
 - Such tools already exist in the **Rodin Platform**

-
- This development required **237 proofs**
 - Initial model: 7
 - 1st refinement: 26
 - 2nd refinement: 66
 - 3rd refinement: 138
 - All proved **automatically** by the Rodin Platform

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

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

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

- For the init event in the initial model

Axioms of the constants \Rightarrow Modified Invariants	INV
---	-----

- For other events in the initial model

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

- This rule is not mandatory

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

- For old events only

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

- For init event only

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

- 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
---	-----

- For new events only

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

- For new events only

Axioms of the constants Abstract invariants Concrete invariants Disj. of abs. guards \Rightarrow Disj. of conc. guards	VAR
---	-----

- Global proof rule

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

- For old events (in case of superposition)

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