17. Train System

Jean-Raymond Abrial

2009

Purpose of Lecture

- To present the development of a a train system controller

- It is a piece of software helping a train agent to control trains

- The goal is to have trains safely circulating in a certain network

Why is it Interesting?

- Studying a very complex data structure: the track network.

- Studying a very complex object: a train on a track network.

- Reliability is absolutely fundamental in this project

- Showing once again the modelling of a closed system
  (software + environment)

- Showing once again a systematic methodology

Outline

- Explaining the problem and constructing the requirement document

- Defining the refinement strategy

- Constructing the formal model (and doing the proofs)

- Conclusion

An Aside: System Engineering at Work

- A real study is done presently on the same subject for RATP

- Size of initial "bad" document: 90 pages

- Time length of the study: 6 months (one engineer)

- Re-writing of the requirement document: 2.5 months

- Development of the formal model and its proofs: 2.5 months

- Writing a final "system study" document, plus an animation: 1 month

System Engineering at Work (cont'd)

- Re-writing of the requirement document:
    - Understanding the problem: 2 weeks
    - Getting more explanation and information: 3 weeks
    - Writing the explanatory text: 4 weeks
    - Writing the reference text (the precise requirements): 2 weeks

- The client (RATP) is required to sign the new requirement document

- Development of the formal model:
    - Refinement strategy and allocation of requirements: 2 weeks
    - Construction and proof of the formal model: 2 months

Back to our Problem: Requirement Document Conventions

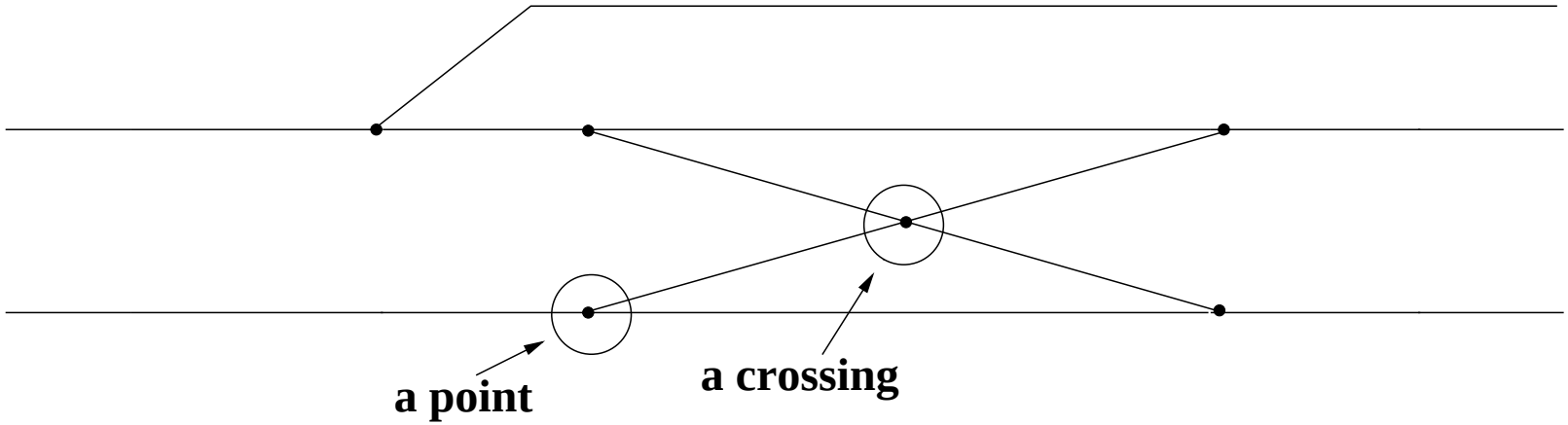
ENV	Environment
FUN	Functional
SAF	Safety

MVT	Movement
TRN	Train
FLR	Failure

The goal of the train system is to safely control trains moving on a track network	FUN-1

Track Network

- Here is a (simplified) track network controlled by a train agent

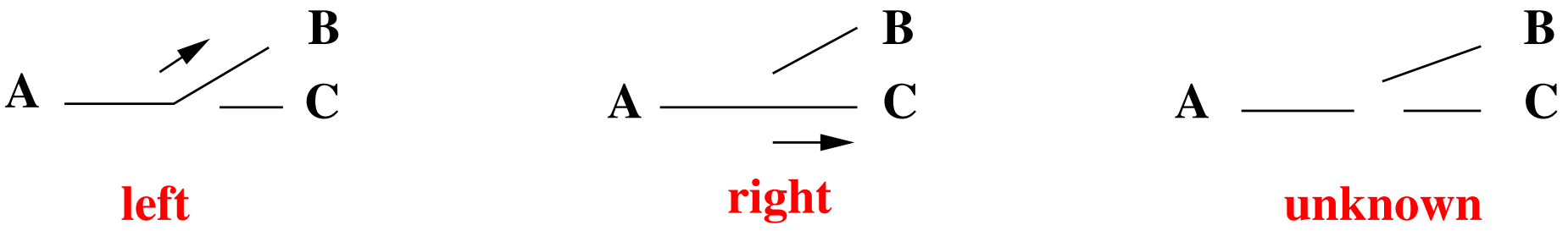


- Two kinds of special components: points and crossings

A track network may contain some special components: points and crossings	ENV-1

Points

- A point might be in three different positions: left, right, and unknown

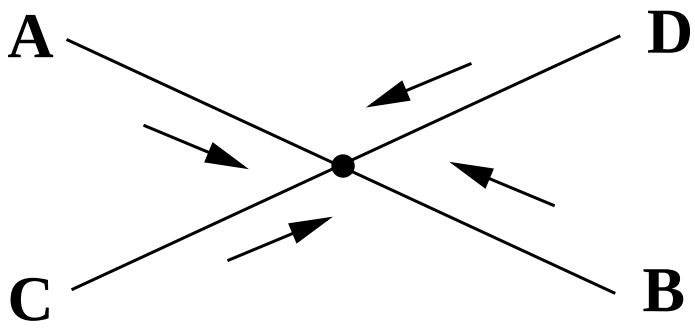


- For simplification, the unknown position is not considered

A point may have two positions: left or right	ENV-2

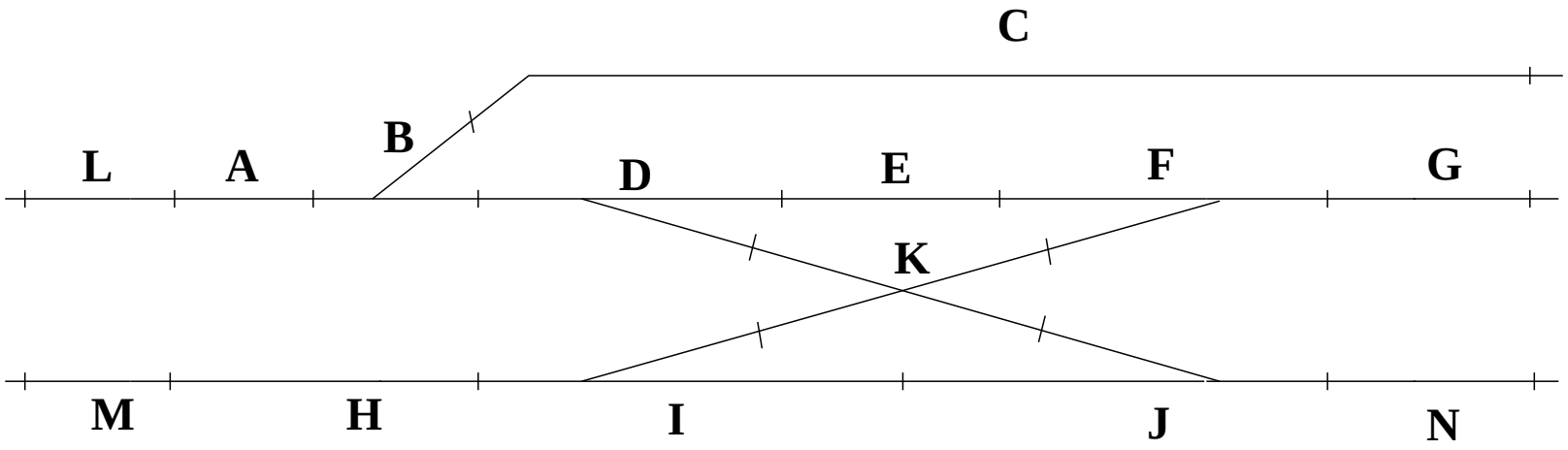
Crossing

- Unlike a point, a crossing has no state



The Concept of Block

A track network is made of a number of named fixed blocks	ENV-3



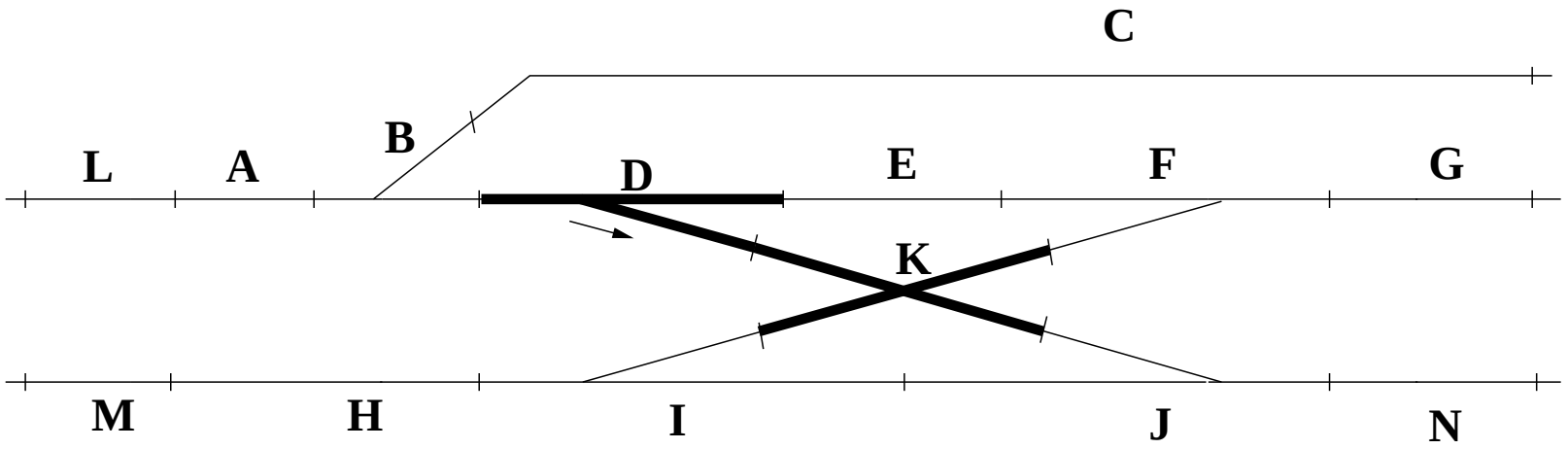
A special component (points or crossings) is always attached to a given block. A block contains at most one special component	ENV-4

Block Occupancy

- Each block is equipped with a track circuit

- It is used to detect the presence of a train on the concerned block

A block may be occupied or unoccupied by a train	ENV-5

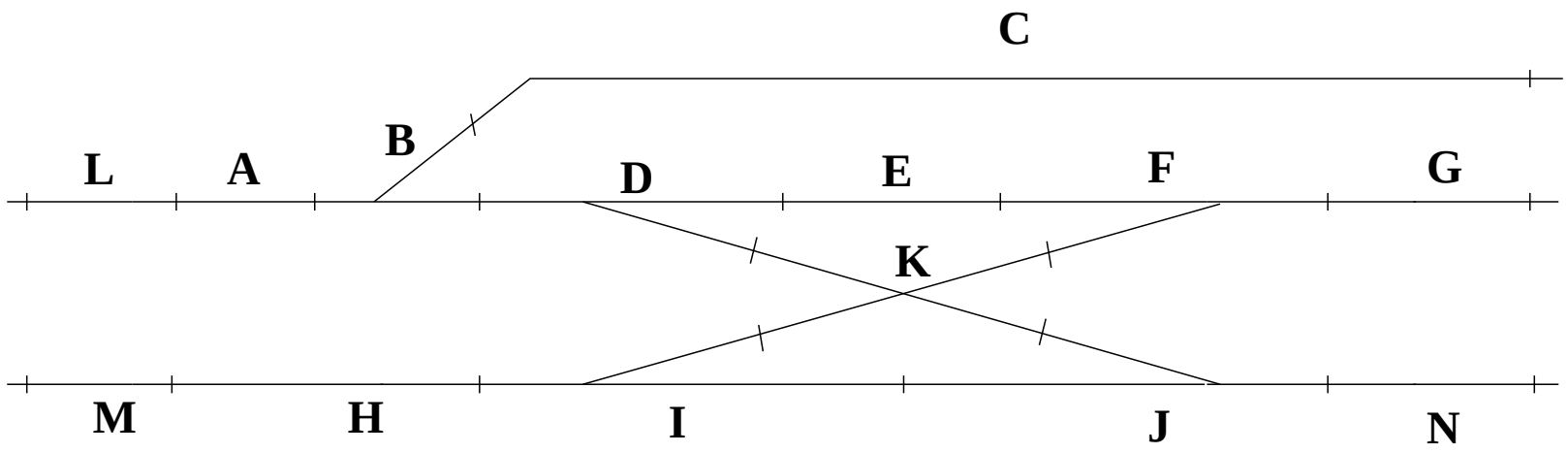


The Concept of Route

A network has a fixed number of routes. Each route is characterized by a sequence of adjacent blocks	ENV-6

- The controller will allow trains to use some routes

Example of Routes

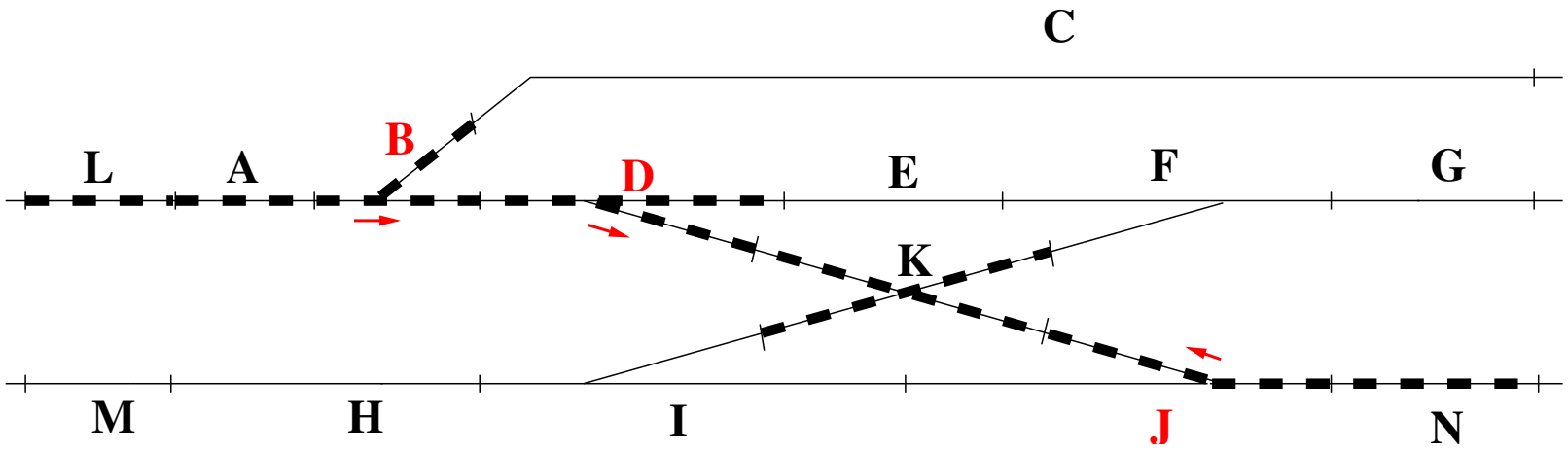


$R1$	$L\ A\ B\ C$	$R6$	$C\ B\ A\ L$
$R2$	$L\ A\ B\ D\ E\ F\ G$	$R7$	$G\ F\ E\ D\ B\ A\ L$
$R3$	$L\ A\ B\ D\ K\ J\ N$	$R8$	$N\ J\ K\ D\ B\ A\ L$
$R4$	$M\ H\ I\ K\ F\ G$	$R9$	$G\ F\ K\ I\ H\ M$
$R5$	$M\ H\ I\ J\ N$	$R10$	$N\ J\ I\ H\ M$

Additional Characterization of Routes

A route is also characterized by the positions of the points which are situated in blocks composing it	ENV-7

- Route $R3$: L A B D K J N



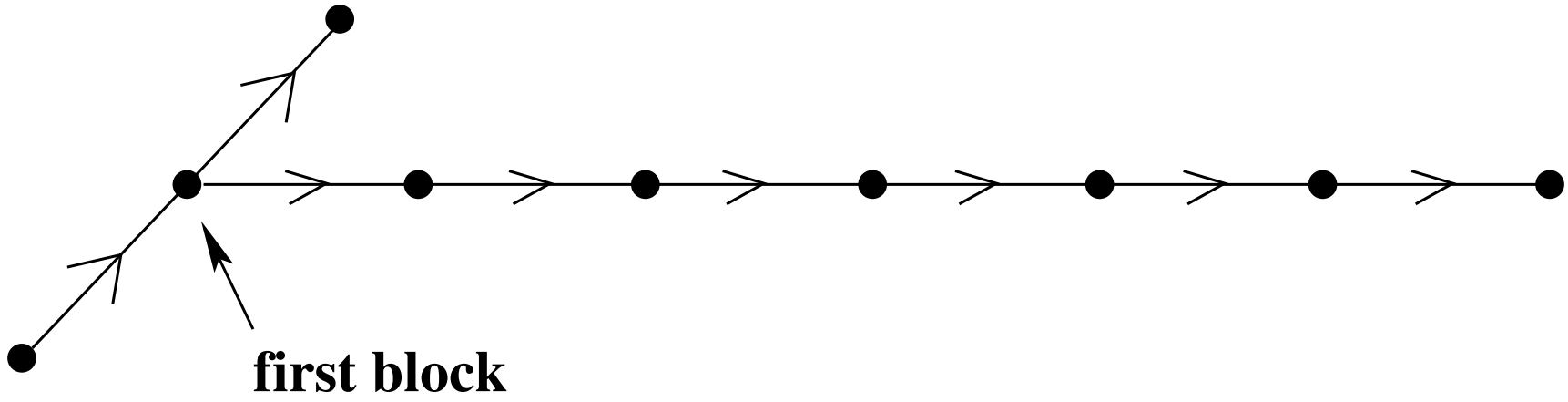
- the point in block B is positioned to the right,

- the point in block D is positioned to the right,

- the point in block J is positioned to the right.

More Properties of a Route

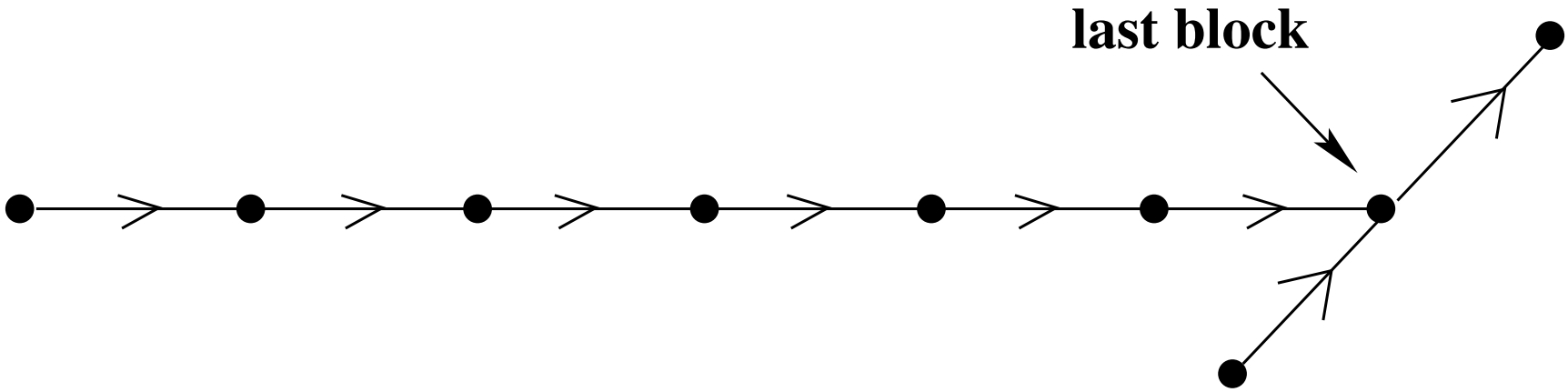
- Not allowed!



The first block of a route cannot be part of another route unless it is also the first or last block of that route	ENV-8

More Properties of a Route (cont'd)

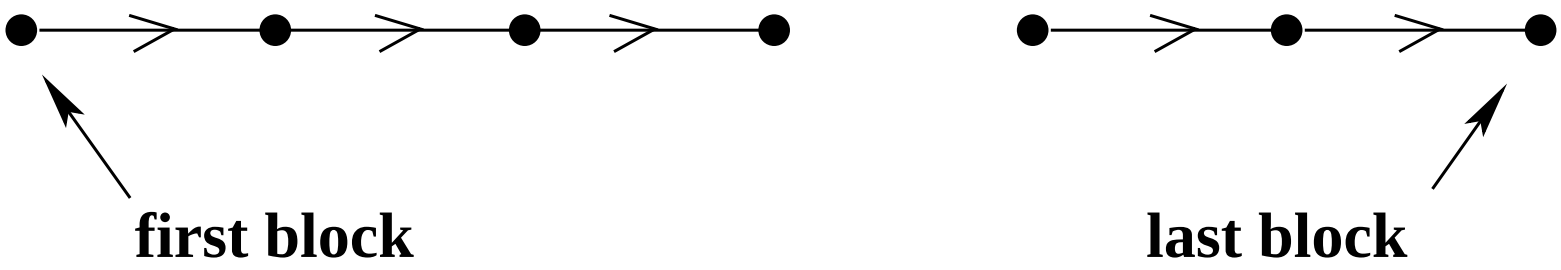
- Not allowed!



The last block of a route cannot be part of another route unless it is also the first or last block of that route	ENV-9

More Properties of a Route (cont'd)

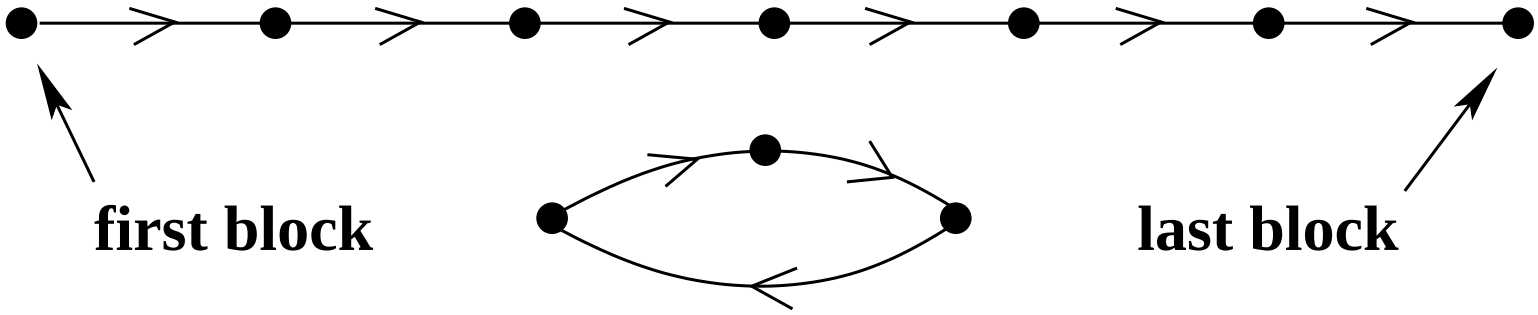
- Not allowed!



A route connects its first block to its last one in a continuous manner	ENV-10

More Properties of a Route (cont'd)

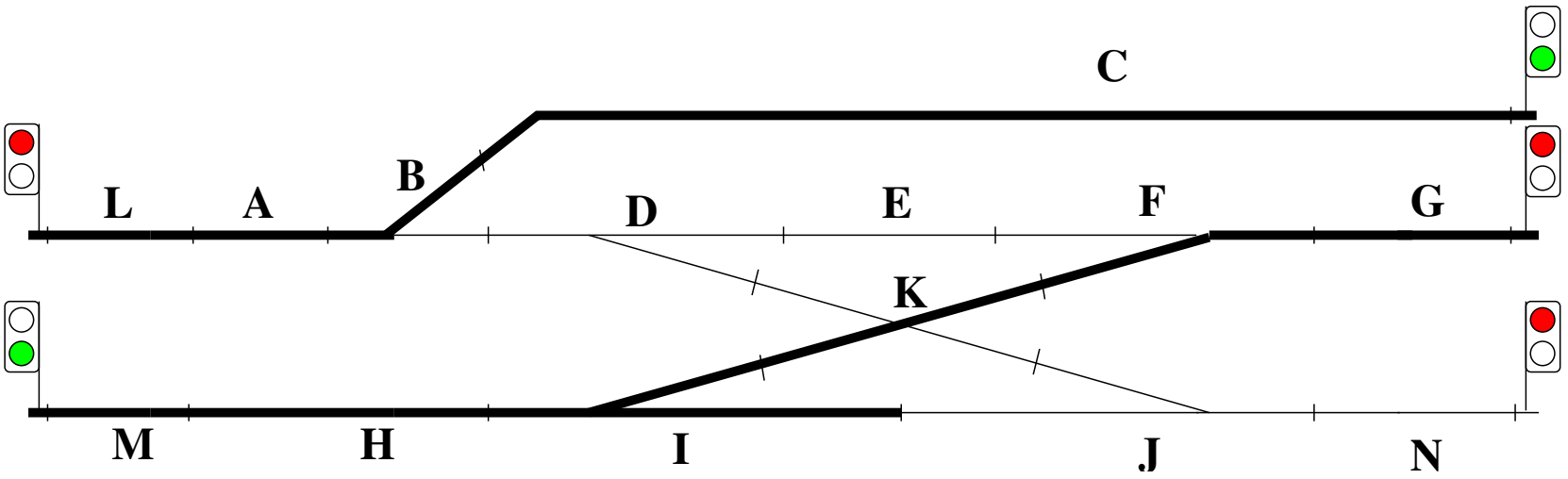
- Not allowed!



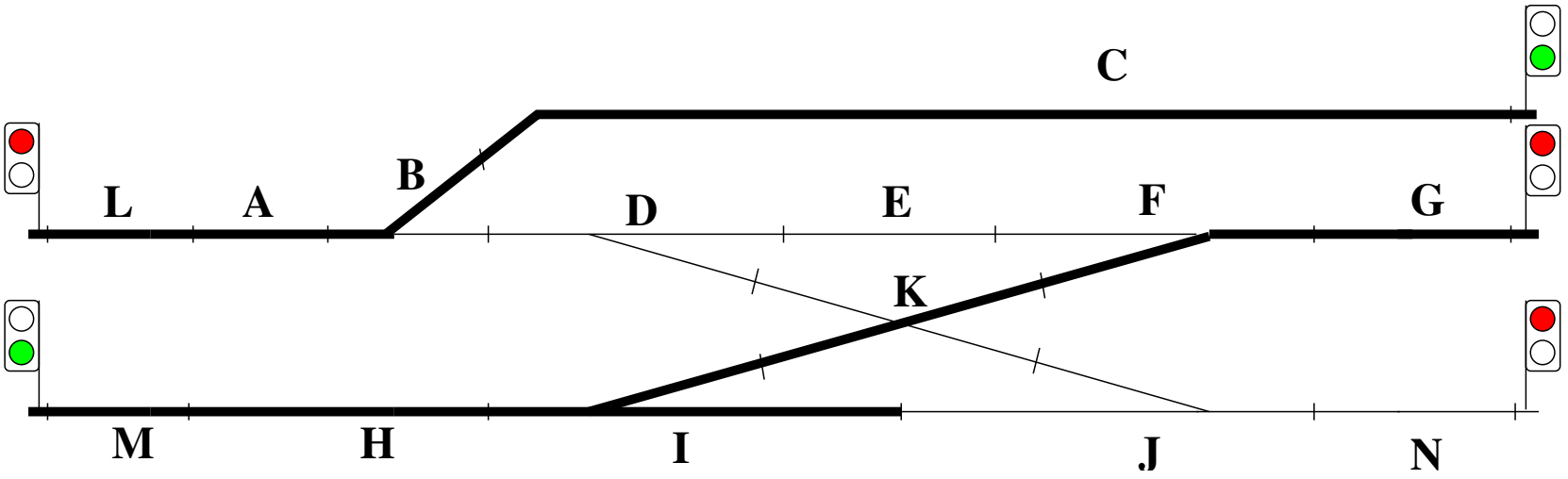
A route contains no cycles	ENV-11

Each route is protected by a signal situated just before its first block	ENV-12

A signal can be red or green. Trains are supposed to stop at red signals	ENV-13



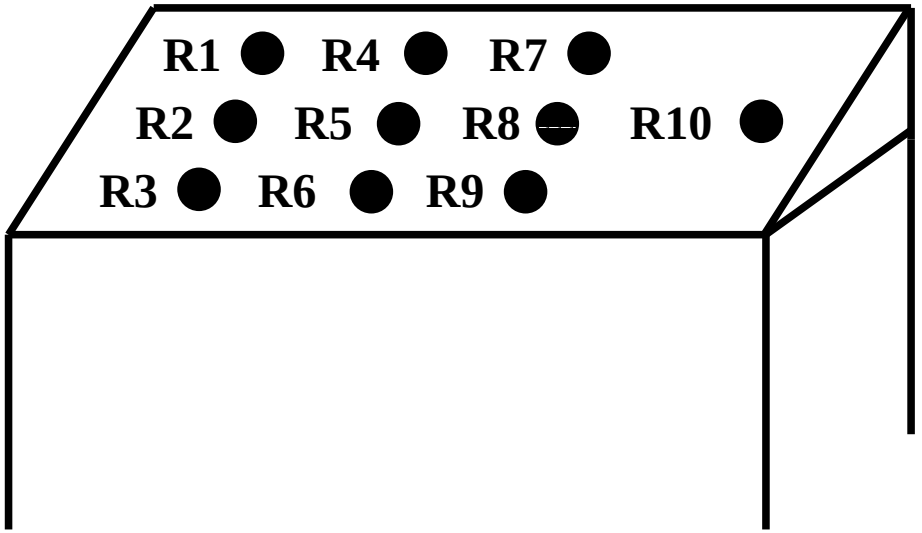
Routes having the same first block share the same signal	ENV-14



A green signal turns back to red automatically as soon as the first block is made occupied	ENV-15

Route and Block Reservation

- The train agent is provided with a panel with route commands.



A route can be reserved for a train. The software is in charge of controlling the reservation process	FUN-2

Reservation Process

- The reservation process of a route r is made of three phases:

1. the individual reservation of the blocks of r,
2. the positioning of the relevant points of r,
3. the turning to green of the signal protecting route r.

Phase 1: Block Reservation

A block can be reserved or free	FUN-3

An occupied block is always reserved	FUN-4

Reserving a route consists in reserving the individual blocks it is made of. Once this is done, the route is said to be reserved	FUN-5

Once it is reserved, a route has to be formed by properly positioning its points	FUN-6

A formed route is always a reserved route	FUN-7

Phase 3:Turning Signal to Green

Once it is formed, a route is made available for the incoming train by turning its signal to green	FUN-8

- Main risks:

    - Two trains traversing the network hit each other.

    - A point changing position under a train.

    - The point of a route changing position in front of a train.

- In all cases, A TRAIN MAY DERAIL

Safety Conditions

A block can be reserved for at most one route	SAF-1

The signal of a route can only be green when all blocks of that route are reserved for it and are unoccupied, and when all points of this route are properly positioned	SAF-2

Safety Conditions (cont'd)

A point can only be re-positioned if it belongs to a block which is in a reserved but not yet formed route	SAF-3

No blocks of a reserved, but not yet formed, route are occupied	SAF-4

Moving Conditions

Once a block of a formed route is made unoccupied, it is also freed	MVT-1

A route remains formed as long as there are some reserved blocks in it	MVT-2

A formed route can be made free (not formed and not reserved any more) when no blocks are reserved for it any more	MVT-3

Train Assumptions

- A freed block cannot be made occupied again by the same train without freeing the concerned route

A train cannot split while in the network	TRN-1

A train cannot move backwards in the network	TRN-2

Train Assumptions (cont'd)

A train cannot enter in the middle of a route. It has to do so through its first block.	TRN-3

A train cannot leave a route without first occupying then freeing all its blocks	TRN-4

Failures

Trains are equipped with the Automatic Train Protection system (ATP), which guarantees that they cannot enter a route guarded by a red signal	FLR-1

Trains are equipped with special bindings, which guarantee that they cannot be mechanically broken.	FLR-2

Failures (cont'd)

The ATP and a slight delay observed by the track circuit guarantee that a train moving backward cannot occupy again a block which has been physically freed.	FLR-3

The risk of a faulty detection of a block occupancy is not treated	FLR-4

The case where a short train derails and leaves its block is not treated.	FLR-5

Requirement Summary

- We have 39 requirements: far more in a real train system!

ENV	Environment	15	MVT	Movement	3
FUN	Functional	8	TRN	Train	4
SAF	Safety	4	FLR	Failure	5

Refinement Strategy

- Logical block and route concepts are formalized.

- Physical blocks are introduced and connected to the logical ones.

- A notion of readiness for a route.

- Introduction of the physical signals.

- Introduction of the abstract points.

- More refinements are needed in order to finalize details.

carrier sets: B, R

constants: $rtbl, nxt$

- Set of blocks: B
- Set of routes: R
- Variable $rtbl$ denotes the routes of a block
- Variables nxt denotes the succession of blocks in each route

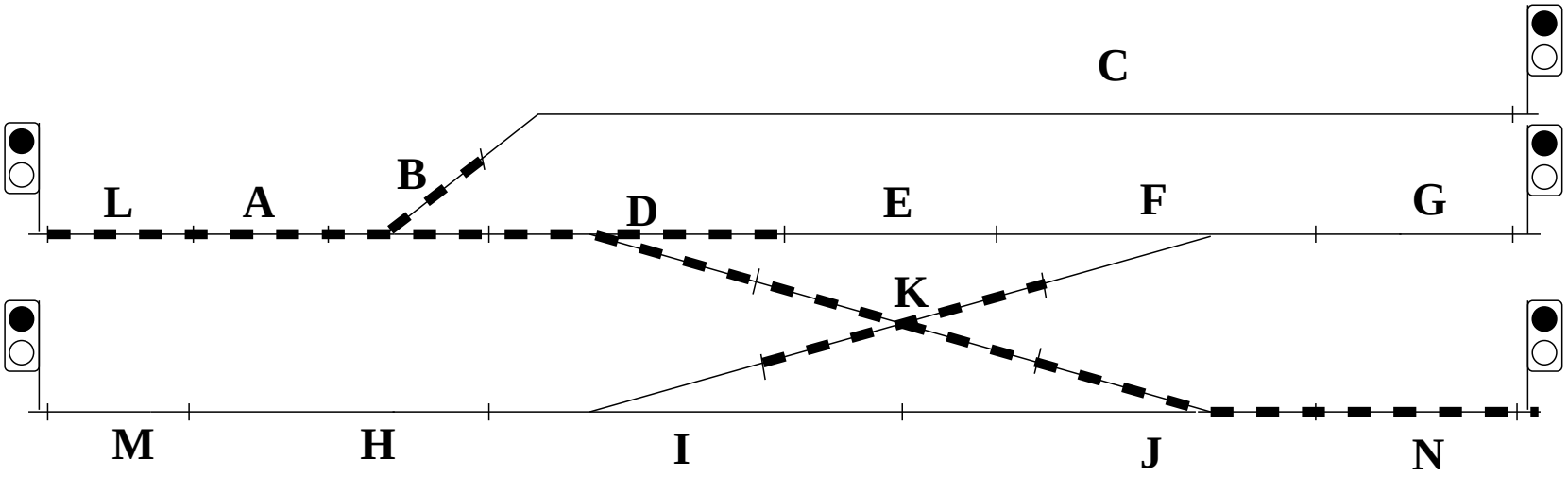
axm0_1: $rtbl \in B \leftrightarrow\!\!\!\!\rightarrow R$

axm0_2: $nxt \in R \rightarrow (B \rightarrowtail\!\!\!\!\!\!\rightarrow B)$

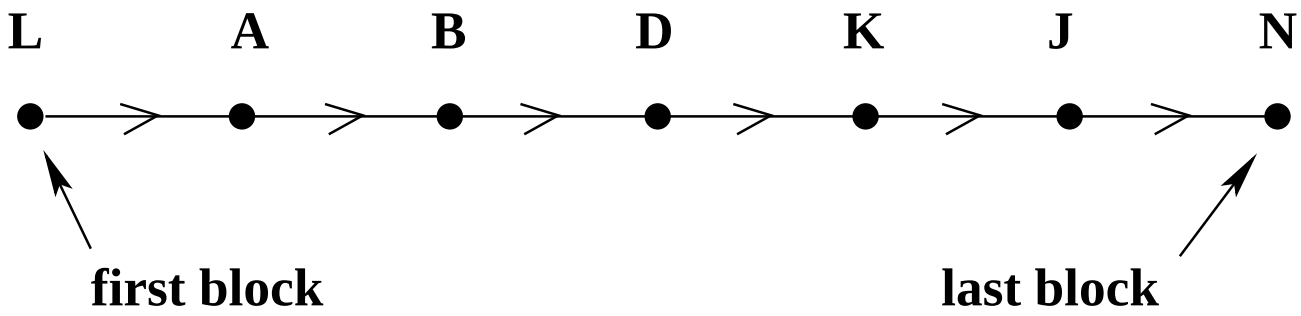
- Notice the injection

Initial Model: Example

- Route $R3$: L A B D K J N



- Here is $nxt(R3)$:



Initial Model: First and Last Block of a Route

constants: $\cdots$ fst, lst

axm0_3: $fst \in R \rightarrow B$

axm0_4: $lst \in R \rightarrow B$

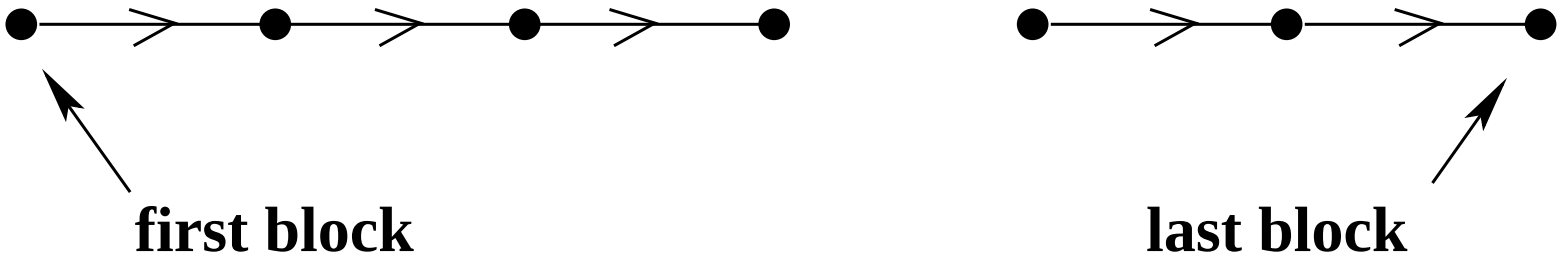
- The first and last block of a route are in that route (**axm0_5**, **axm0_6)**
- They are distinct (**axm0_7**)

axm0_5: $fst^{-1} \subseteq rtbl$

axm0_6: $lst^{-1} \subseteq rtbl$

axm0_7: $\forall r \cdot (r \in R \Rightarrow fst(r) \neq lst(r))$

- To be avoided!

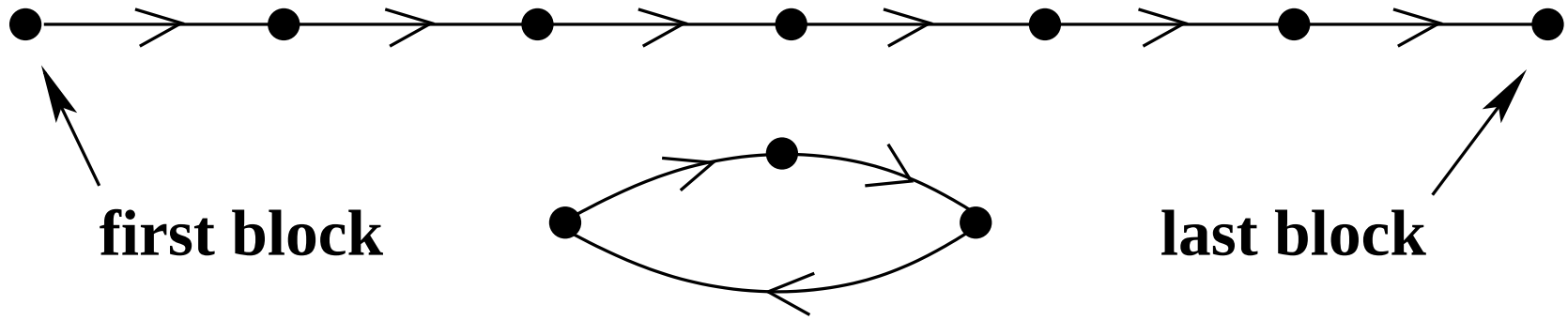


axm0_8: $\forall r \cdot \left(\begin{array}{l} r \in R \\ \Rightarrow \\ nxt(r) \in s \setminus \{lst(r)\} \rightarrowtail\!\!\!\!\!\rightarrow s \setminus \{fst(r)\} \end{array} \right)$

where s is $rtbl^{-1}[\{r\}]$ (the blocks of route r)

Continuity (cont'd)

- To be avoided!



axm0_9: $$\forall r \cdot \left(r \in R \Rightarrow \forall S \cdot \left(\begin{array}{l} S \subseteq B \\ S \subseteq nxt(r)[S] \\ \Rightarrow \\ S = \varnothing \end{array} \right) \right)$$

Beginning and End of Routes

- The first block of a route r cannot be the block of another route s unless if first or last of s

axm0_10: $$\forall r, s \cdot \left(\begin{array}{l} r \in R \\ s \in R \\ r \neq s \\ \Rightarrow \\ fst(r) \notin rtbl^{-1}[\{s\}] \setminus \{fst(s), lst(s)\} \end{array} \right)$$

Beginning and End of Routes (cont'd)

- The last block of a route r cannot be the block of another route s unless if first or last of s

axm0_11: $$\forall r, s \cdot \left(\begin{array}{l} r \in R \\ s \in R \\ r \neq s \\ \Rightarrow \\ lst(r) \notin rtbl^{-1}[\{s\}] \setminus \{fst(s), lst(s)\} \end{array} \right)$$

Initial Model: Variables

variables: $resrt$, $resbl$, $rsrtbl$, OCC

inv0_1: $resrt \subseteq R$

inv0_2: $resbl \subseteq B$

inv0_3: $rsrtbl \in resbl \rightarrow resrt$

inv0_4: $rsrtbl \subseteq rtbl$

inv0_5: $OCC \subseteq resbl$

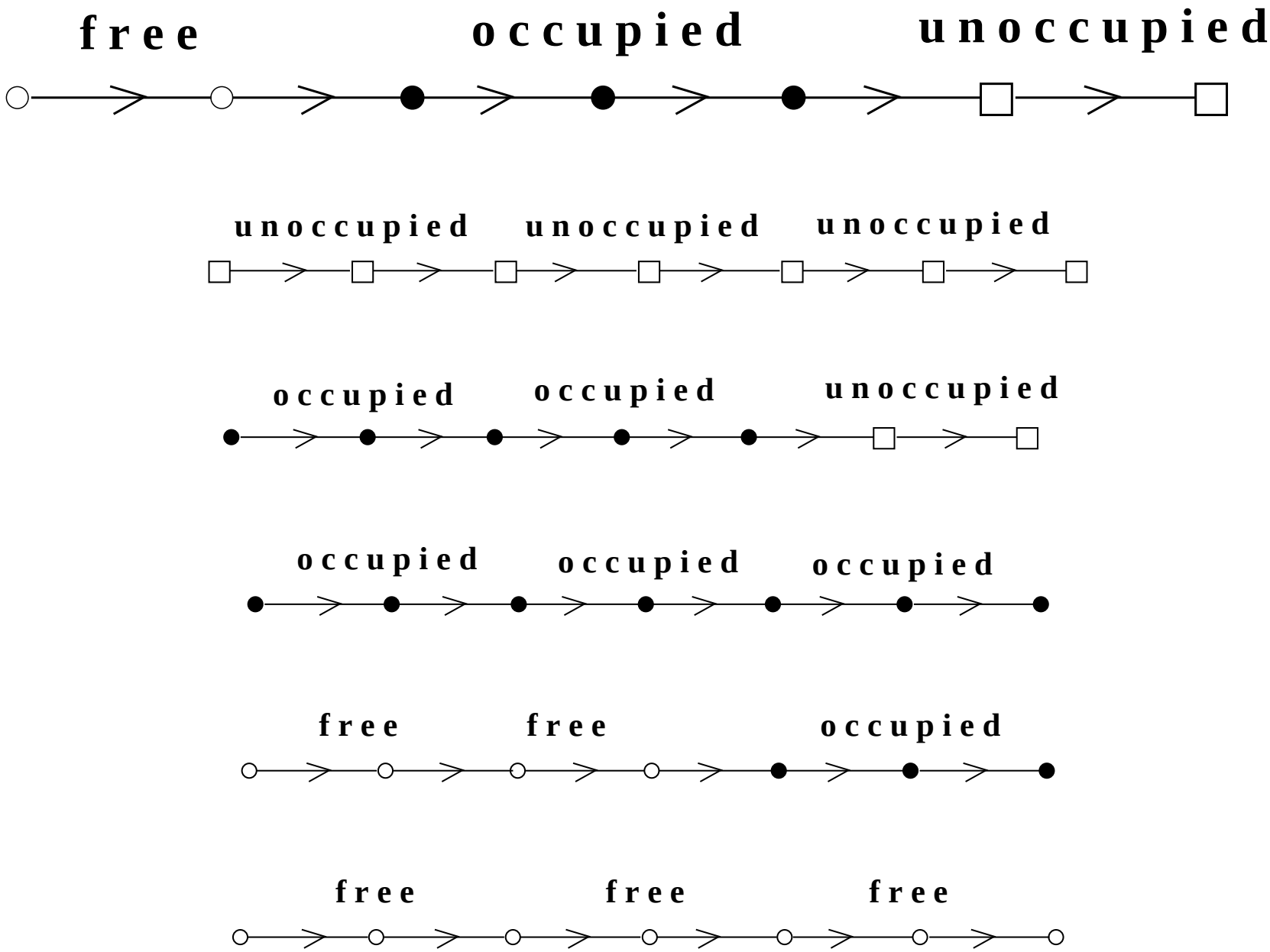
- Set of reserved routes: $resrt$, a controller variable

- Set of reserved blocks: $resbl$, a controller variable

- Reserved route of reserved block: $rsrtbl$, a controller variable

- Set of occupied block: OCC, an environment variable (UPPER case)

Train in a Reserved Route (General Case and Special Cases)



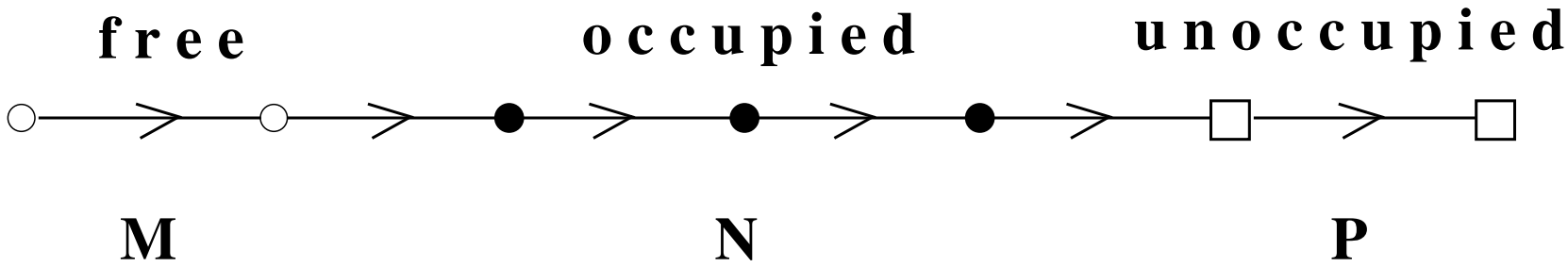
Given a reserved route r:

- Let M be the free blocks of r (those behind the train)
- Let N be the occupied blocks in r (those where the train is)
- Let P be the unoccupied blocks in r (those in front of the train)

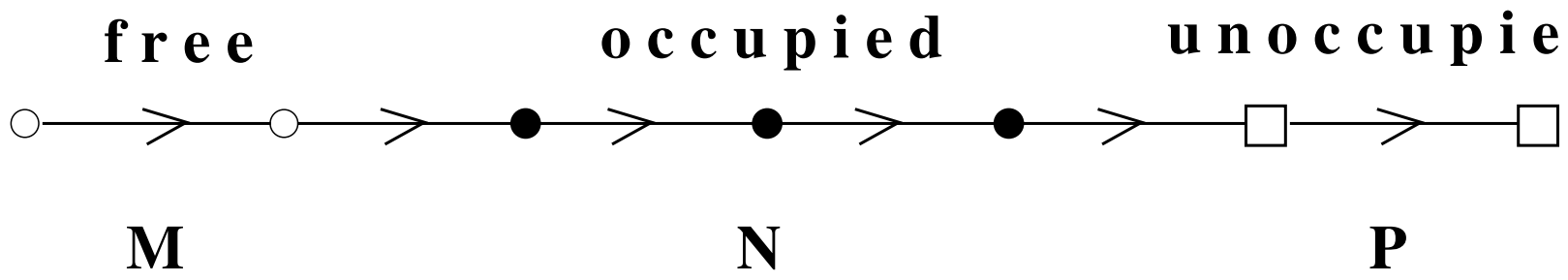
$$M = rtbl^{-1}[\{r\}] \setminus rsrtbl^{-1}[\{r\}]$$

$$N = rsrtbl^{-1}[\{r\}] \cap OCC$$

$$P = rsrtbl^{-1}[\{r\}] \setminus OCC$$



Formalizing Trains in Reserved Routes (cont'd)



- Here are the possible transitions: $MMMMNNNNNPPPP$

$$M \to M \qquad M \to N \qquad N \to N \qquad N \to P \qquad P \to P$$

Formalizing Trains in Reserved Routes (cont'd)

$$M \to M \quad M \to N \quad N \to N \quad N \to P \quad P \to P$$

$$nxt(r)[M] \subseteq M \cup N \qquad nxt(r)[N] \subseteq N \cup P \qquad nxt(r)[P] \subseteq P$$

- Such conditions are equivalent to the following ones

$$nxt(r)[M] \cap P = \varnothing \quad nxt(r)[N \cup P] \subseteq N \cup P \quad nxt(r)[P] \subseteq P$$

inv0_6: $\forall r \cdot (r \in R \Rightarrow nxt(r)[rtbl^{-1}[\{r\}] \setminus s] \cap (s \setminus OCC) = \varnothing)$

inv0_7: $\forall r \cdot (r \in R \Rightarrow nxt(r)[s] \subseteq s)$

inv0_8: $\forall r \cdot (r \in R \Rightarrow nxt(r)[s \setminus OCC] \subseteq s \setminus OCC)$

where s is $rsrtbl^{-1}[\{r\}]$

Initial Models: Events

- Controller events:
    - route_reservation,
    - route_freeing.

- Environment events:
    - FRONT_MOVE_1,
    - FRONT_MOVE_2,
    - BACK_MOVE.

Controller Events: route_reservation

- r is a non-reserved route
- No block of r is reserved

route_reservation

any r **where**

$r \in R \setminus resrt$

$rtbl^{-1}[\{r\}] \cap resbl = \varnothing$

then

$resrt := resrt \cup \{r\}$

$rsrtbl := rsrtbl \cup rtbl \rhd \{r\}$

$resbl := resbl \cup rtbl^{-1}[\{r\}]$

end

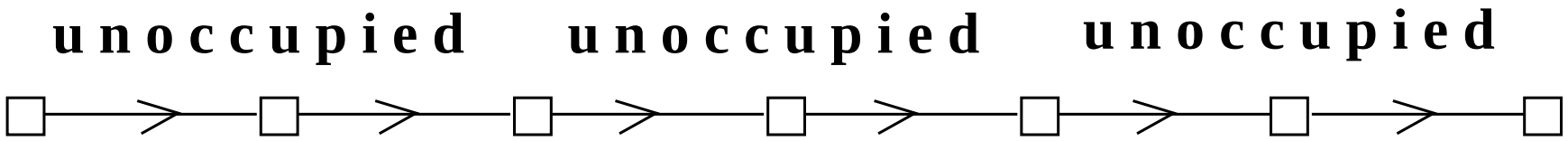
Controller Events (cont'd)

- r is a reserved route
- r has no block reserved for it any more

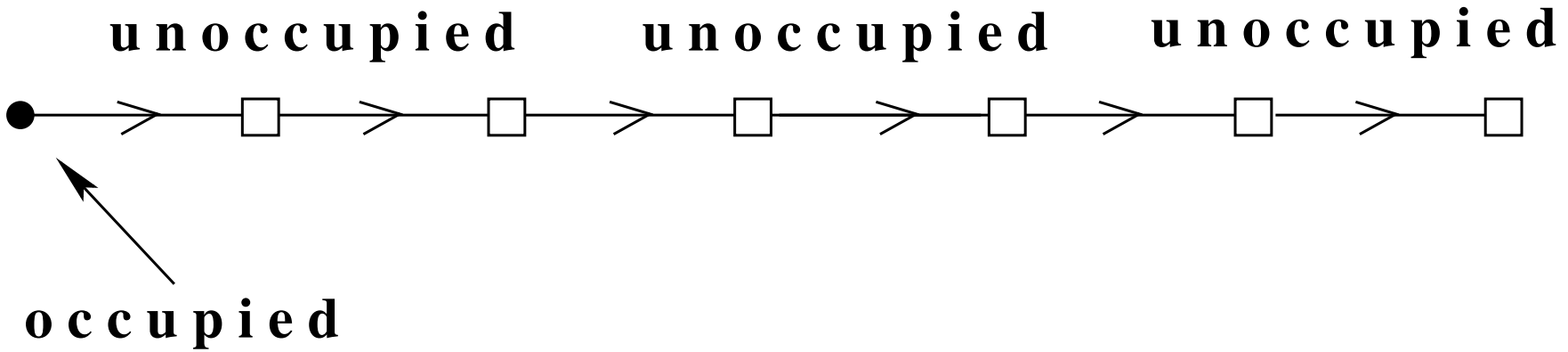
route_freeing
  any r **where**
    $r \in resrt$
    $r \notin \mathrm{ran}(rsrtbl)$
  then
    $resrt := resrt \setminus \{r\}$
  end

Environment Events: Arrival of a train in a route r

- Before:



- After:



Environment Events: Arrival of a train in a route r

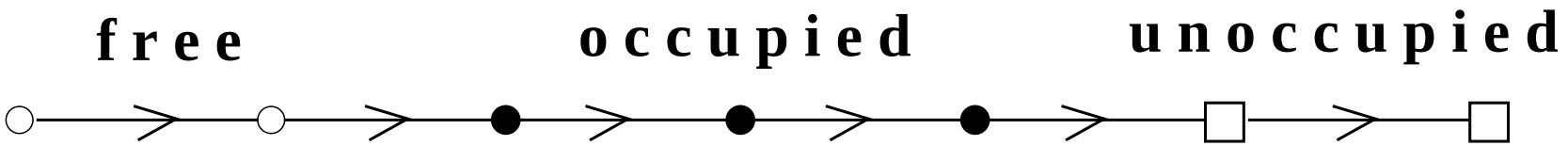
- r is a reserved route
- $fst(r)$ is a reserved and unoccupied block
- The reserved route of $fst(r)$ is indeed r

FRONT_MOVE_1
any r **where**
$r \in resrt$
$fst(r) \in resbl \setminus OCC$
$rsrtbl(fst(r)) = r$
then
$OCC := OCC \cup \{fst(r)\}$
end

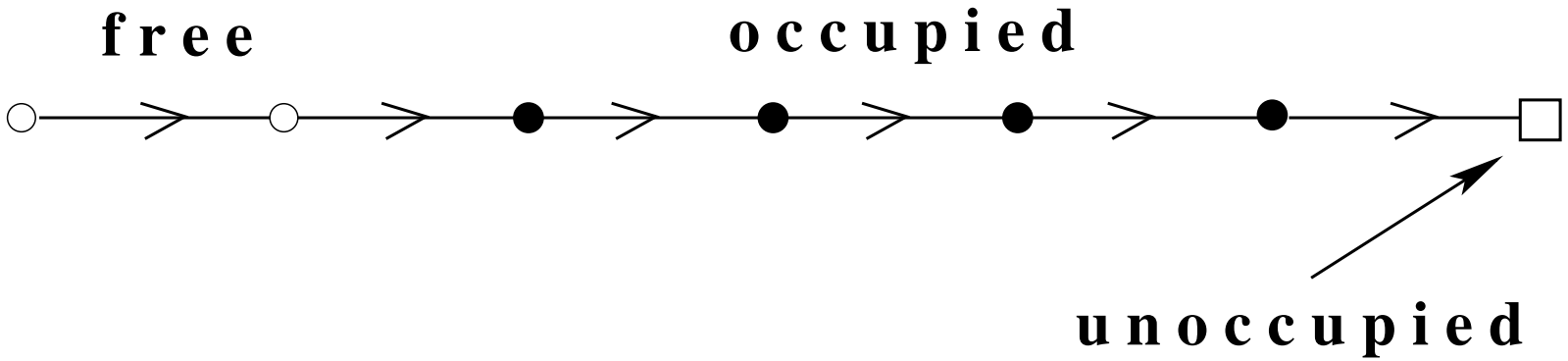
- The guard depends on some controller variables

Environment Events: Move of the front of a train

- Before:



- After:



Environment Events: Move of the front of a train

- b is an occupied block
- c is an unoccupied block
- c is next to b in the reserved route of b

FRONT_MOVE_2

any b, c **where**

$b \in OCC$

$c \in B \setminus OCC$

$b \mapsto c \in nxt(rsrtbl(b))$

then

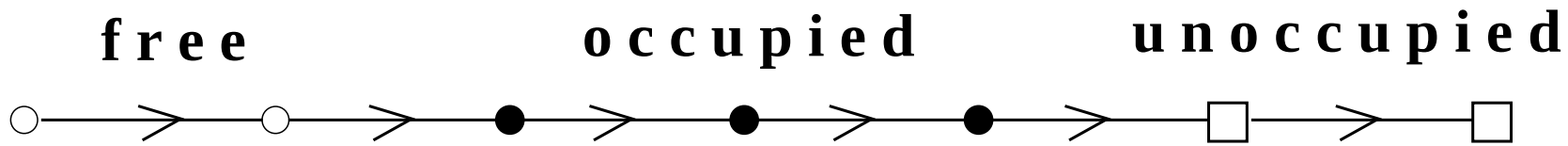
$OCC := OCC \cup \{c\}$

end

- The guard depends on some controller variables

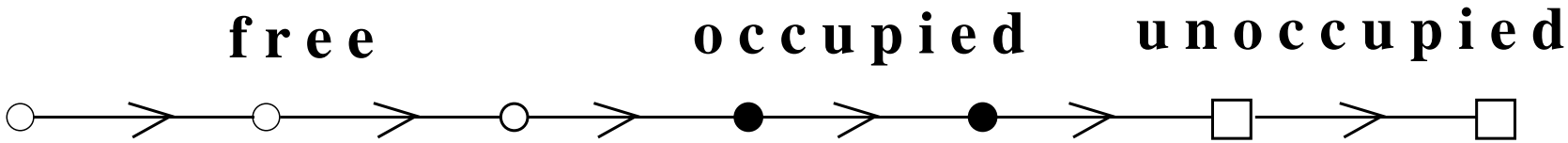
Environment Events: Move of the Back of a Train

- Before:



- "free" blocks behind the train can be reserved for another route

- After:



Environment Events: Move of the Back of a Train

- b is an occupied block

- if any, the block next to b is also occupied

- if any, the block before b is not reserved or reserved for a route not equal to that of b

BACK_MOVE
  any b, n **where**
    $b \in OCC$
    $n = nxt(rsrtbl(b))$
    $b \in \mathrm{dom}(n) \Rightarrow n(b) \in OCC$
    $b \in \mathrm{ran}(n) \Rightarrow n^{-1}(b) \notin \mathrm{dom}(rsrtbl) \lor rsrtbl(n^{-1}(b)) \neq rsrtbl(b)$
  then
    $OCC := OCC \setminus \{b\}$
    $rsrtbl := \{b\} \lhd rsrtbl$
    $resbl := resbl \setminus \{b\}$
  end

- The guard depends on some controller variables

First Refinement: Variables

variables: $\cdots$, TRK, frm, LBT

- Variable TRK denotes the physical succession of blocks

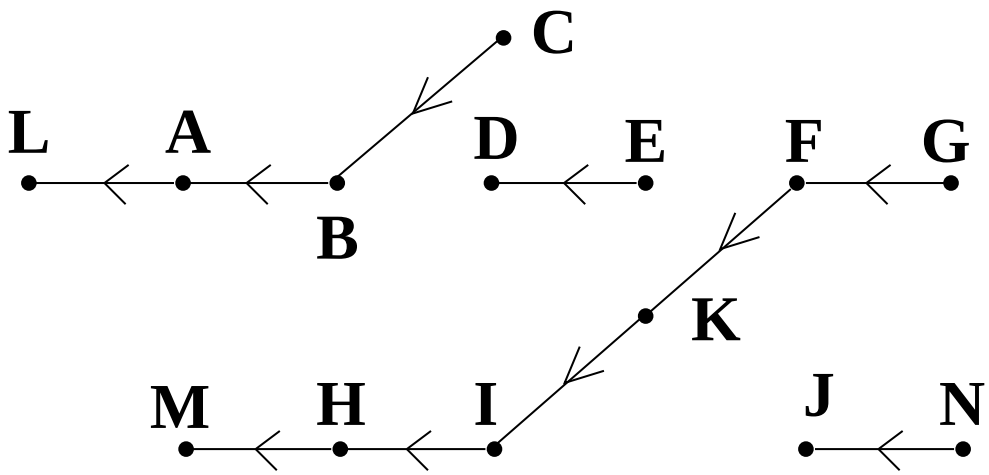
- Variable frm denotes the set of formed routes

- Variable LBT denotes the set of last blocks occupied by trains

Variable TRK

inv1_1: $TRK \in B \rightarrowtail B$

- Here is an illustration of the variable TRK in a certain situation:



- Route $R9$ (G F K I H H M) and $R6$ (C B A L) are visible.

- The crossing in block K is "broken"

- The physical track "remembers" the direction followed by trains

Variable frm

- A formed route is a reserved route
- Routes of occupied blocks are formed
- A reserved but not formed route has all its blocks reserved

inv1_3: $frm \subseteq resrt$

inv1_4: $rsrtbl[OCC] \subseteq frm$

inv1_5: $\forall r \cdot \left(\begin{array}{l} r \in resrt \setminus frm \\ \Rightarrow \\ rtbl \rhd \{r\} = rsrtbl \rhd \{r\} \end{array} \right)$

Variable frm: the Most Important Invariant

- Let r be a formed route

- The logical succession of reserved blocks of r

$$rsrtbl^{-1}[\{r\}] \lhd nxt(r)$$

- is identical to the physical succession of reserved blocks of r

$$rsrtbl^{-1}[\{r\}] \lhd TRK$$

inv1_6: $$\forall r \cdot \left(\begin{array}{l} r \in frm \\ \Rightarrow \\ rsrtbl^{-1}[\{r\}] \lhd nxt(r) = rsrtbl^{-1}[\{r\}] \lhd TRK \end{array} \right)$$

- In other words, the points are well positioned

Variable LBT

- The last block of a train is occupied

inv1_7: $LBT \subseteq OCC$

If the last block b of a train is preceded by a block then that block is either not reserved or reserved for a route different from that of b

inv1_8: $\forall b \cdot \left(\begin{array}{l} b \in LBT \\ b \in \mathrm{ran}(n) \\ \Rightarrow \\ n^{-1}(b) \notin \mathrm{dom}(rsrtbl) \quad \vee \quad rsrtbl(n^{-1}(b)) \neq rsrtbl(b) \end{array} \right)$

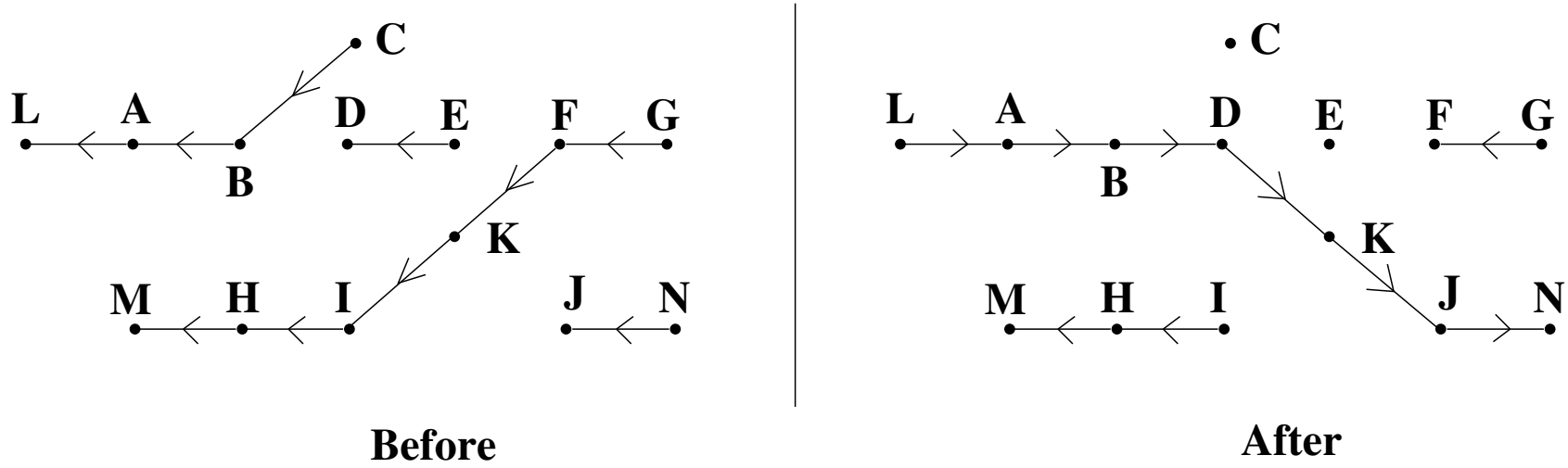
where n is $nxt(rsrtbl(b))$

First Refinement: Events

- Controller events:
    - point_positioning, (new event)
    - route_formation (new event)
    - route_reservation,
    - route_freeing,

- Environment events:
    - FRONT_MOVE_1,
    - FRONT_MOVE_2,
    - BACK_MOVE_1, (split)
    - BACK_MOVE_1 (split)

Event point_positioning



- An abstract view of point positioning:

point_positioning
any r **where**
$r \in resrt \setminus frm$
then
$TRK := (\mathrm{dom}(nxt(r)) \lhd TRK \rhd \mathrm{ran}(nxt(r))) \cup nxt(r)$
end

- The physical track of a reserved but not yet formed route is modified

Event route_formation

- r is a reserved but not yet formed route

- the logical succession of blocks of r coincides with the physical one

route_formation
  any r **where**
    $r \in resrt \setminus frm$
    $rsrtbl^{-1}[\{r\}] \lhd nxt(r) = rsrtbl^{-1}[\{r\}] \lhd TRK$
  then
    $frm := frm \cup \{r\}$
  end

- This corresponds to a response from the physical track

- The controller verifies that the points are well positioned

Arrival of a new train on a route

(abstract-)FRONT_MOVE_1
any r **where**
$r \in resrt$
$fst(r) \in resbl \setminus OCC$
$rsrtbl(fst(r)) = r$
then
$OCC := OCC \cup \{fst(r)\}$
end

- The (abstract) train still has access to some controller variables

FRONT_MOVE_1
any r **where**
$r \in frm$
$fst(r) \in resbl \setminus OCC$
$rsrtbl(fst(r)) = r$
then
$OCC := OCC \cup \{fst(r)\}$
$LBT := LBT \cup \{fst(r)\}$
end

Front move of a train

(abstract-)FRONT_MOVE_2
any b, c **where**
$b \in OCC$
$c \in B \setminus OCC$
$b \mapsto c \in nxt(rsrtbl(b))$
then
$OCC := OCC \cup \{c\}$
end

The train now follows the physical track

FRONT_MOVE_2
any b **where**
$b \in OCC$
$b \in \mathrm{dom}(TRK)$
$TRK(b) \notin OCC$
then
$OCC := OCC \cup \{TRK(b)\}$
end

A Train Leaves its Route

(abstract-)BACK_MOVE
any b, n **where**
$b \in OCC$
$n = nxt(rsrtbl(b))$
$b \in \mathrm{dom}(n) \Rightarrow n(b) \in OCC$
$\left(\begin{array}{l} b \in \mathrm{ran}(n) \ \wedge \\ n^{-1}(b) \in \mathrm{dom}(rsrtbl) \\ \Rightarrow \\ rsrtbl(n^{-1}(b)) \neq rsrtbl(b) \end{array} \right)$
then
$OCC := OCC \setminus \{b\}$
$rsrtbl := \{b\} \lhd rsrtbl$
$resbl := resbl \setminus \{b\}$
end

BACK_MOVE_1
any b **where**
$b \in LBT$
$b \notin \mathrm{dom}(TRK)$
then
$OCC := OCC \setminus \{b\}$
$rsrtbl := \{b\} \lhd rsrtbl$
$resbl := resbl \setminus \{b\}$
$LBT := LBT \setminus \{b\}$
end

- The train follows the physical track

- A message is sent and treated by the controller (later split into 2 events)

Back Move of a Train

(abstract-)BACK_MOVE

any b, n **where**

$b \in OCC$

$n = nxt(rsrtbl(b))$

$b \in \mathrm{dom}(n) \Rightarrow n(b) \in OCC$

$$\left(\begin{array}{l} b \in \mathrm{ran}(n) \wedge \\ n^{-1}(b) \in \mathrm{dom}(rsrtbl) \\ \Rightarrow \\ rsrtbl(n^{-1}(b)) \neq rsrtbl(b) \end{array} \right)$$

then

$OCC := OCC \setminus \{b\}$

$rsrtbl := \{b\} \lhd rsrtbl$

$resbl := resbl \setminus \{b\}$

end

BACK_MOVE_2

any b **where**

$b \in LBT$

$b \in \mathrm{dom}(TRK)$

$TRK(b) \in OCC$

then

$OCC := OCC \setminus \{b\}$

$rsrtbl := \{b\} \lhd rsrtbl$

$resbl := resbl \setminus \{b\}$

$LBT := (LBT \setminus \{b\}) \cup \{TRK(b)\}$

end

- The train follows the physical track

- A message is sent and treated by the controller (later split into 2 events)

Second Refinement: Variables

variables: $\cdots$, rdy

inv2_1: $rdy \subseteq frm$

inv2_2: $\forall r \left(\begin{array}{l} r \in rdy \\ \Rightarrow \\ rtbl \rhd \{r\} = rsrtbl \rhd \{r\} \end{array} \right)$

inv2_3: $\forall r \left(\begin{array}{l} r \in rdy \\ \Rightarrow \\ \mathrm{dom}(rtbl \rhd \{r\}) \cap OCC = \varnothing \end{array} \right)$

Event route_formation

route_formation

any r **where**

$r \in resrt \setminus frm$

$rsrtbl^{-1}[\{r\}] \lhd nxt(r) = rsrtbl^{-1}[\{r\}] \lhd TRK$

then

$frm := frm \cup \{r\}$

$rdy := rdy \cup \{r\}$

end

Arrival of a new train on a route

(abstract-)FRONT_MOVE_1
any r **where**
$r \in frm$
$fst(r) \in resbl \setminus OCC$
$rsrtbl(fst(r)) = r$
then
$OCC := OCC \cup \{fst(r)\}$
$LBT := LBT \cup \{fst(r)\}$
end

This event still uses some controller variables (wait until next refinement!)

FRONT_MOVE_1
any r **where**
$r \in rdy$
$rsrtbl(fst(r)) = r$
then
$OCC := OCC \cup \{fst(r)\}$
$LBT := LBT \cup \{fst(r)\}$
$rdy := rdy \setminus \{r\}$
end

Third refinement: Introducing Signals

- Signals implement the concept of readiness (new carrier set S)

carrier sets: B, R, S

constants: $\cdots$ SIG

axm3_1: $SIG \in \mathrm{ran}(fst) \rightarrowtail\!\!\!\!\!\!\rightarrow S$

variables: $\cdots$ GRN

inv3_1: $GRN \subseteq S$

inv3_2: $SIG[fst[rdy]] = GRN$

Event route_formation

route_formation

any r **where**

$r \in resrt \setminus frm$

$rsrtbl^{-1}[\{r\}] \lhd nxt(r) = rsrtbl^{-1}[\{r\}] \lhd TRK$

then

$frm := frm \cup \{r\}$

$GRN := GRN \cup \{SIG(fst(r))\}$

end

Arrival of a new train on a route

(abstract-)FRONT_MOVE_1
any r **where**
$r \in rdy$
$rsrtbl(fst(r)) = r$
then
$OCC := OCC \cup \{fst(r)\}$
$LBT := LBT \cup \{fst(r)\}$
$rdy := rdy \setminus \{r\}$
end

- The train now follows the physical track and obeys the green signal

FRONT_MOVE_1
any b **where**
$b \in \mathrm{dom}(SIG)$
$SIG(b) \in GRN$
then
$OCC := OCC \cup \{b\}$
$LBT := LBT \cup \{b\}$
$GRN := GRN \setminus \{SIG(b)\}$
end

More Refinements

- More Refinements are needed in order to:
    - Introduce the physical points
    - Decompose events route_reservation, route_formation, and point_positioning in more atomic events

Summary of Proofs

	Number of proofs	Automatic	Interactive
Initial Model	40	24	16
1st Refinement	46	26	20
2nd Refinement	26	15	11
3rd Refinement	12	9	3
4th Refinement	10	8	2
Total	134	82	52

- Not so many proofs (134), but a large proportion of interactive proofs (39%)

- Some interactive proofs are complex