Formal Methods && Tools Group
IEI-CNUCE (ISTI) - Area della Ricerca CNR - Via G. Moruzzi, 1 - 56124 PISA

SNCF Model Reduction

Identification of a subset of SNCF statecharts and Rules

FORMAL METHODS && TOOLS GROUP
ISTI – CNR
Area della Ricerca CNR
Via G. Moruzzi,1
I-56124 - Pisa
Tel. 050.3152920 - Fax 050.3152810

Revisions

Revision	Data	Nome	Pages	
	18 December 2004		42	Preliminary version

Revision	1
File Name	SNCF model reduction.doc

	Name	Signature	Date
Written	Michele Banci		18 December 2004
Approved			

Index

INDEX OF FIGURES

1 Introduction

1.1 Definitions

1.2 Purpose of the document

This report describes the reduction applied to the SNCF specification in order to identify a closed set of statecharts able to performs a complete interlocking function.

The work was finalized to identify the subset of statecharts doing some assumptions. These assumptions have permitted to minimize the variables used into a statechart, but that need of another statecharts in order to elaborate the values of them. This kind of feature has been named “nesting”, in fact when a statechart has got a variable, there is the necessity to calculate its value, for this reason another statecharts has to be evaluated (“nesting”).

The assumptions done and the complete subset is presented starting form next section.

FIRST LEVEL:

GRAPHE_ETAT_IT

2 GRAPHE_ETAT_IT (route state graph)

2.1 Graph description

The graph belongs to the activity chart allocated to the management of routes.

This is the first graph studied, because it is the starting point to the route setting; following the terms of this graph the model has been identified.

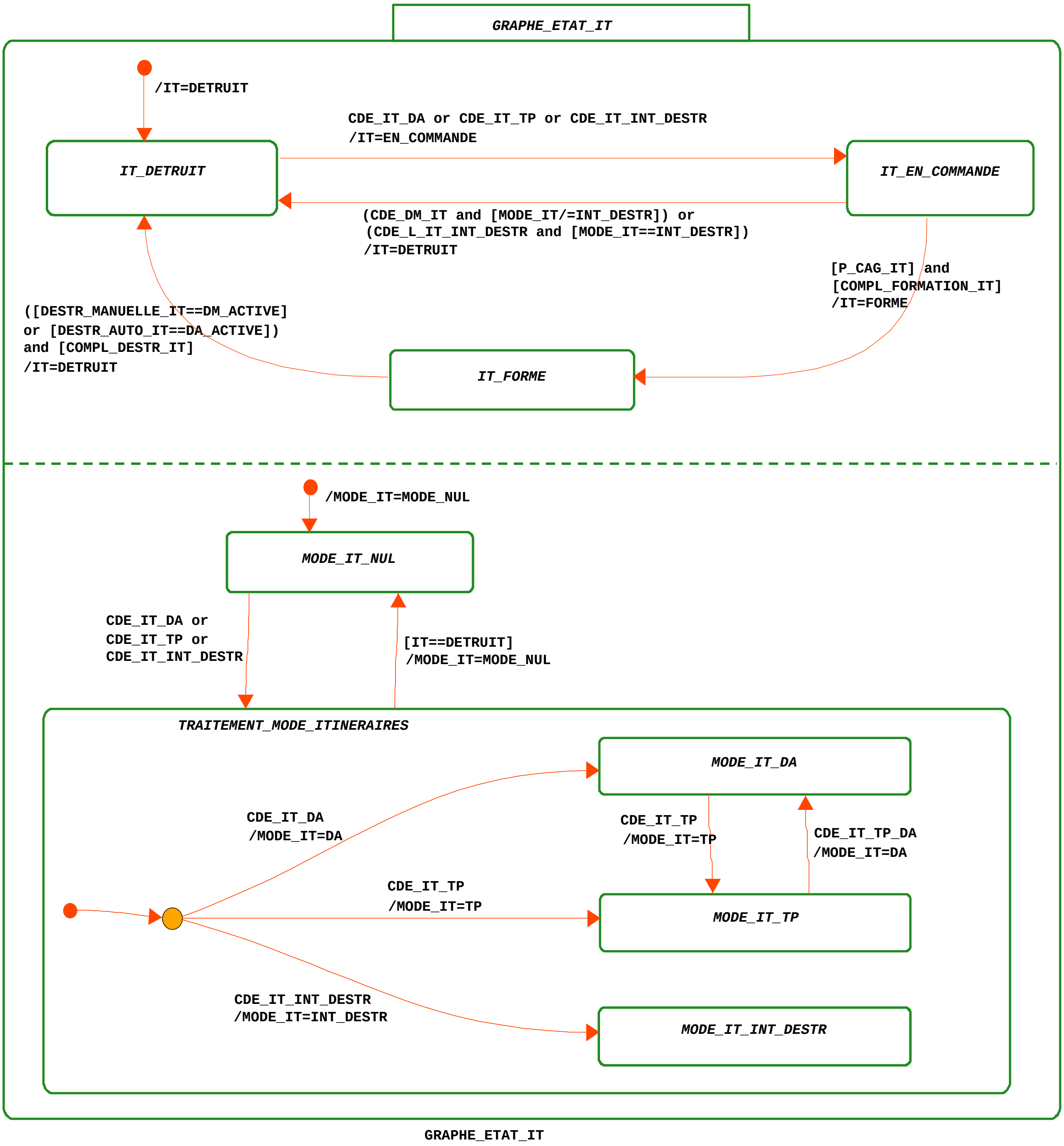


Figure 1 – Graphe_etat_it

2.2 Graph behaviour description

It is split into two “AND” subgraphs, which are independent each other. The graph below permits to choose the kind of route that is permitted: DA=Automatic Destruction, TP=Permanent route (tracé permanent), INT_DESTR=route with destruction not allowed (itinéraires formés avec interdiction de destruction).

Referring to the idea explained above we have reduced the behaviour of the system, and obviously of this graph, to only one mode of route setting, the DA mode (automatic destruction).

For this reason the graph is not reduced, modifying terms of it, but simply not considering the external commands: CMD_IT_TP and CMD_IT_INT_DESTR. We consider only the command CMD_IT_DA.

It is evident that some parts of the graph might not be used (see table 1).

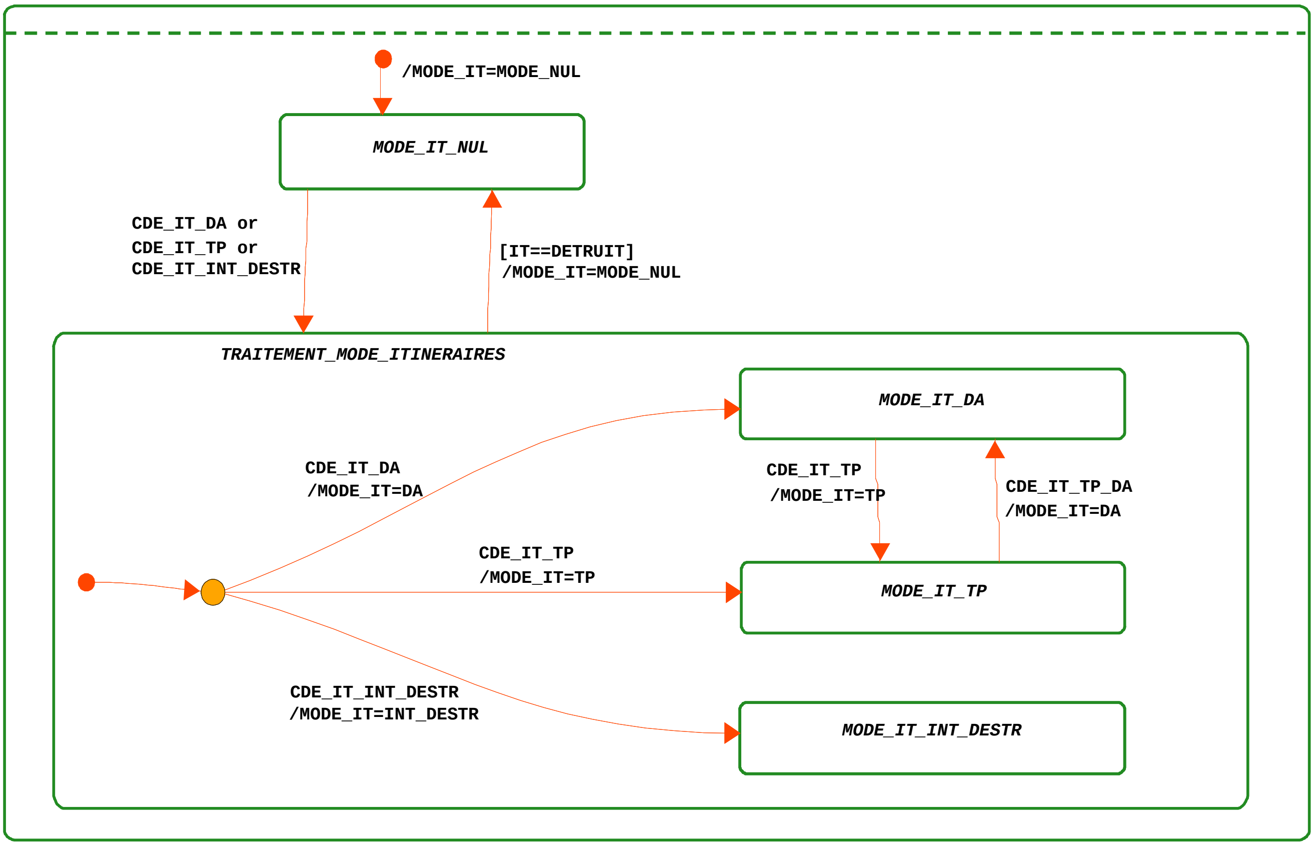


Figure 2 – choice of the kind of route

This part of the graph contains the following elements:

Command inputs	
CMD_IT_DA	used
CMD_IT_TP	not used
CMD_IT_INT_DESTR	not used
Variables	
IT	used

And referring to the second AND-part of the graph:

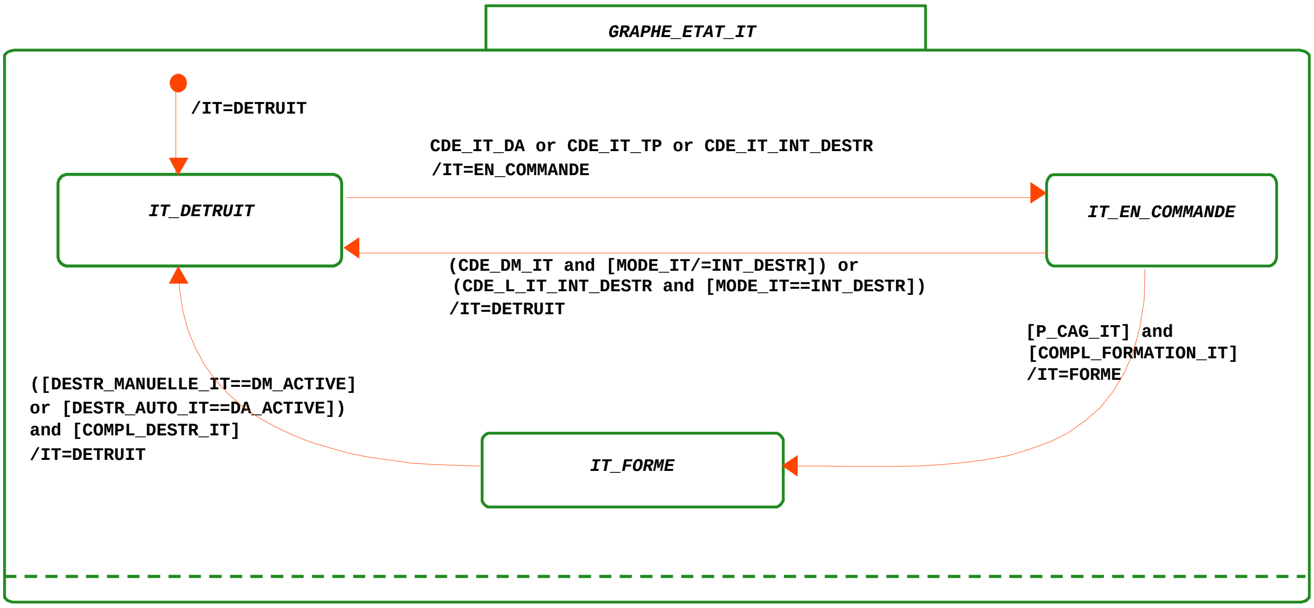

GRAPHE_ETAT_IT
/IT=DETRUIT
CDE_IT_DA or CDE_IT_TP or CDE_IT_INT_DESTR
/IT=EN_COMMANDE
IT_DETRUIT
IT_EN_COMMANDE
(CDE_DM_IT and [MODE_IT/=INT_DESTR]) or
(CDE_L_IT_INT_DESTR and [MODE_IT==INT_DESTR])
/IT=DETRUIT
[P_CAG_IT] and
[COMPL_FORMATION_IT]
/IT=FORME
([DESTR_MANUELLE_IT==DM_ACTIVE]
or [DESTR_AUTO_IT==DA_ACTIVE])
and [COMPL_DESTR_IT]
/IT=DETRUIT
IT_FORME

Figure 3 – route formation and destruction

In Figure 3 is shown the main part of the graph:

Receiving a command of route setting (DA type only) the graph goes to IT_EN_COMMANDE. From this state to IT_FORME it is needed to evaluate two terms that have to be expanded according to the Expansion Rules.

2.2.1 Expansion Rules

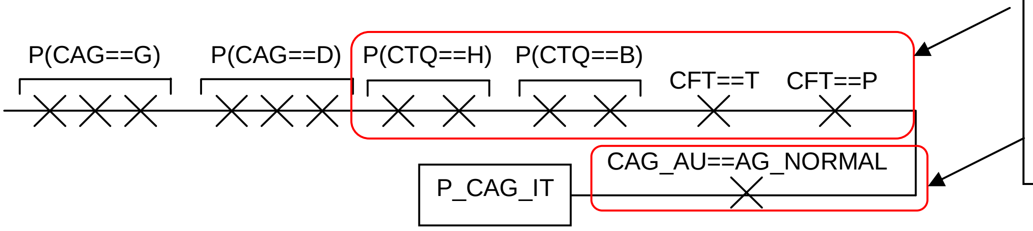
2.2.1.1 COMPL_DESTR_IT

Term always considered “true”.

2.2.1.2 P_CAG_IT

It represents a Boolean equation that is true if all the devices used by this route are in the correct position.

The rule has been reduced eliminating some terms.



- Taquets protegeant l'itineraire.
- Fractionnement
- Aiguille fictive

Figure 4 – Original rule

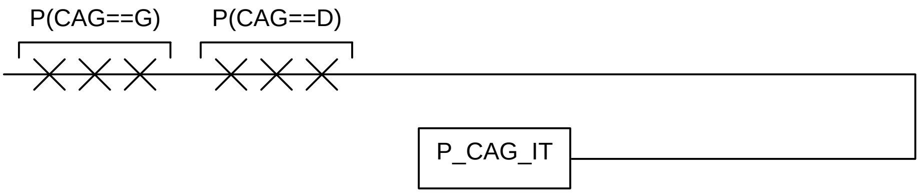


Figure 5 – reduced rule

We consider only the variables related to switch points; the other devices are dealt with in the same way, so they have been eliminated.

P(CAG==G): it is the logical product (AND) of switch points in normal position (Left)

P(CAG==D): it is the logical product (AND) of switch points in reverse position (Right)

These variables are related to other graphs (first level of nesting)

2.2.1.3 COMPL_FORMATION_IT (Complementary conditions for route setting)

It represents a Boolean equation that is true if all the signals over this route have a correct aspect.

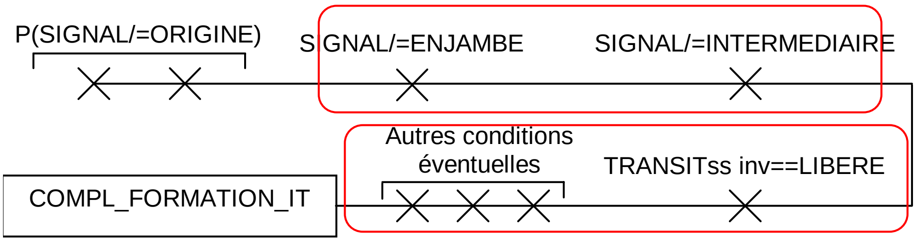


Figure 6 – original rule

The reduction of this rule is the following: The variables have been deleted because they seem to be not clearly explained and also because they do not supply any interesting features to the expansion methodology.

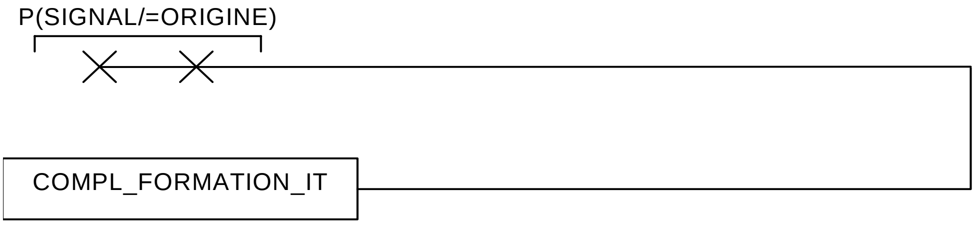
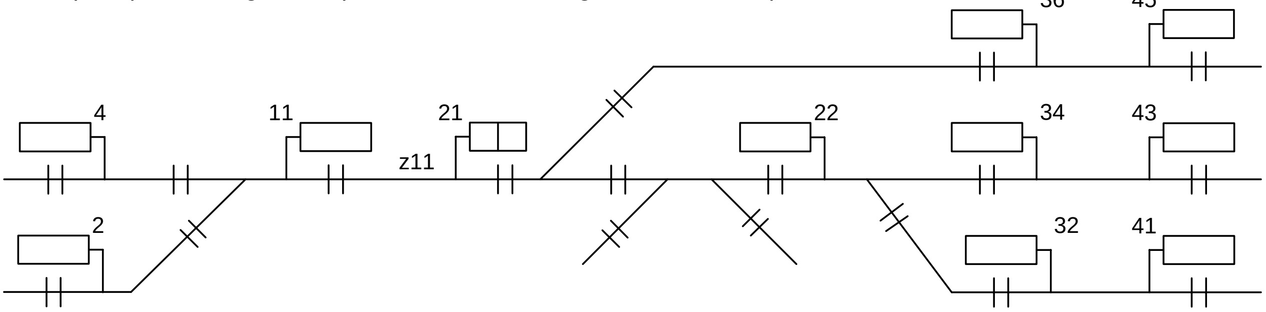


Figure 7 – reduced rule

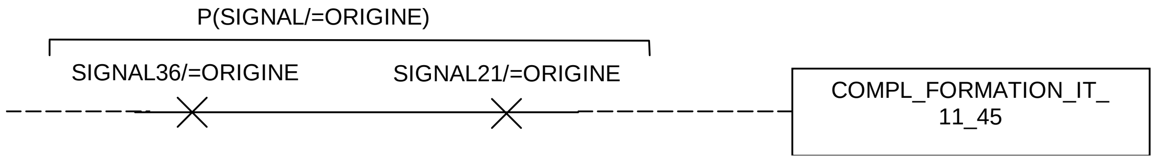
The SIGNAL variables are related to a graph named GRAPHE_ETAT_SIGNAL (first level of nesting)

Examples:

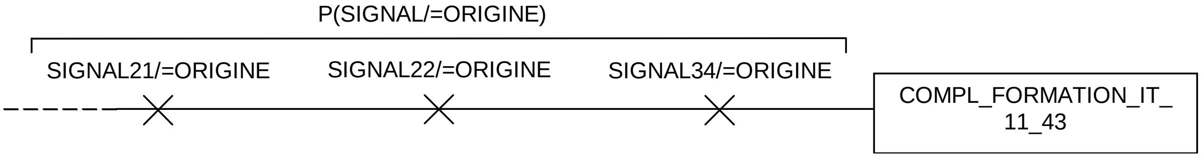
Exemples (tous les signaux représentés ici étant origines d'itinéraires) :



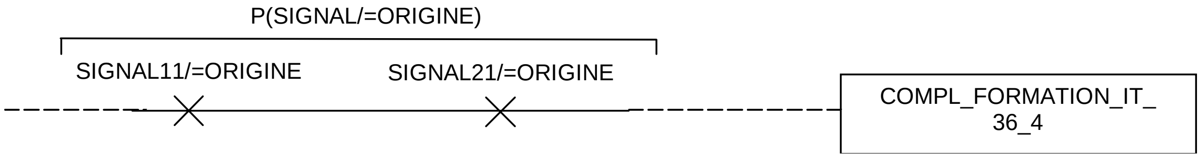
Condition complémentaire de formation de l'itinéraire 11-45 :



Condition complémentaire de formation de l'itinéraire 11-43 :



Condition complémentaire de formation de l'itinéraire 36-4 :



In order to command the route 11_45 the state of the signals 36 and 21 has to be evaluated.

Whether the state of one of these signal is ORIGINE it means that there is another already route set.

Output to the environment:

IT

MODE_IT

Variables stimulating statecharts nesting:

Variable name	***Term name***	***Needed Statechart***
CAG	P_CAG_IT	COMMANDE_AG
SIGNAL	COMPL_FORMATION_IT	ETAT_SIGNAL
DESTR_AUTO_IT		DA_IT_AVEC_DPG

SECOND LEVEL:

GRAPHE_COMMANDE_AG

GRAPHE_ETAT_SIGNAL

GRAPHE_DA_IT_AVEC_DPG

3 GRAPHE_COMMANDE_AG (switch point commander graph)

3.1 Graph description

The graph belongs to the activity chart allocated to the management of route devices.

This is the second graph studied, we have reached this graph because of the variable CAG, following the terms P_CAG_IT of the GRAPHE_ETAT_IT.

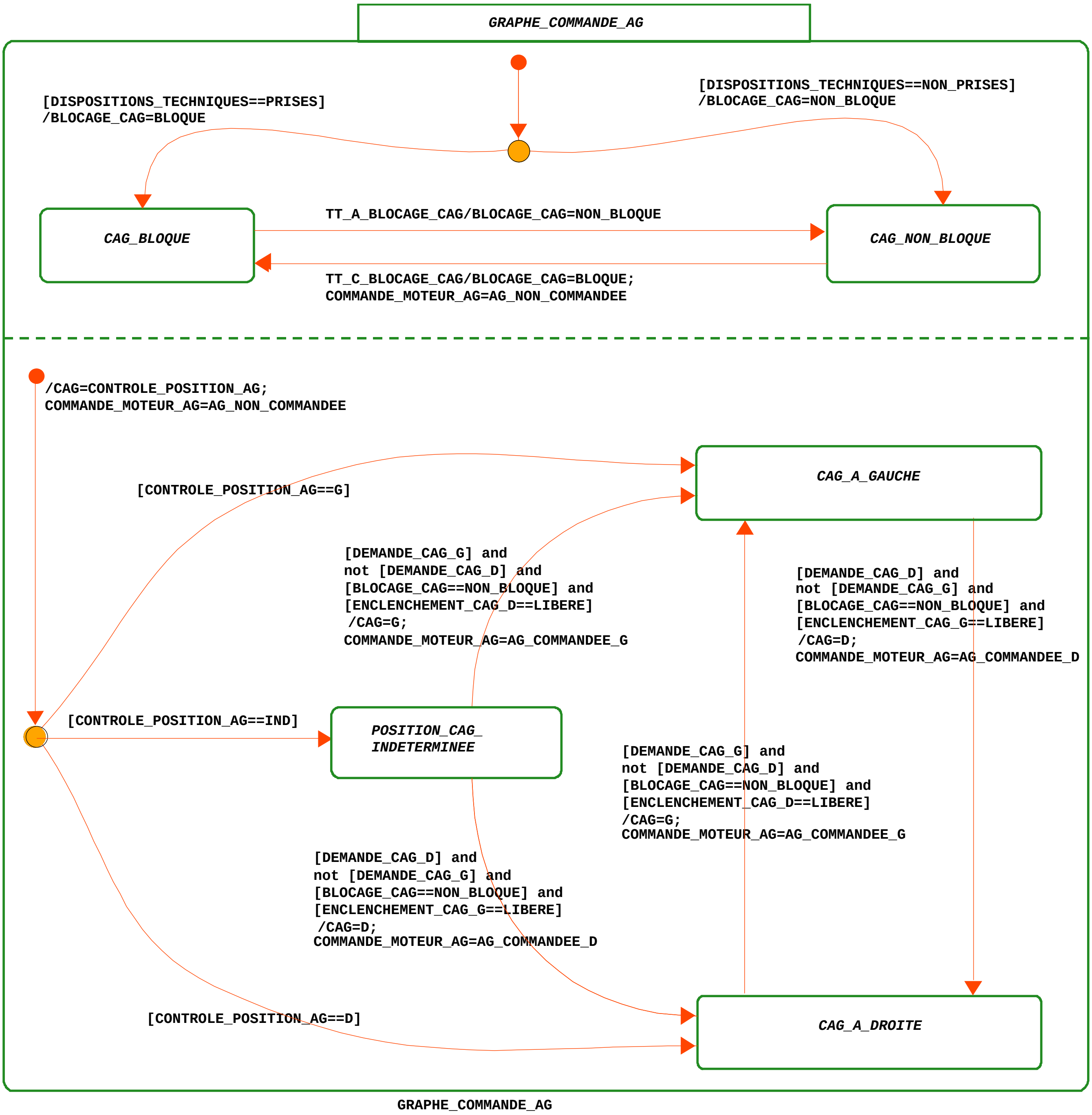

GRAPHE_COMMANDE_AG
[DISPOSITIONS_TECHNIQUES==PRISES]
/BLOCAGE_CAG=BLOQUE
[DISPOSITIONS_TECHNIQUES==NON_PRISES]
/BLOCAGE_CAG=NON_BLOQUE
CAG_BLOQUE
TT_A_BLOCAGE_CAG/BLOCAGE_CAG=NON_BLOQUE
CAG_NON_BLOQUE
TT_C_BLOCAGE_CAG/BLOCAGE_CAG=BLOQUE;
COMMANDE_MOTEUR_AG=AG_NON_COMMANDEE
/CAG=CONTROLE_POSITION_AG;
COMMANDE_MOTEUR_AG=AG_NON_COMMANDEE
[CONTROLE_POSITION_AG==G]
CAG_A_GAUCHE
[DEMANDE_CAG_G] and
not [DEMANDE_CAG_D] and
[BLOCAGE_CAG==NON_BLOQUE] and
[ENCLENCHEMENT_CAG_D==LIBERE]
/CAG=G;
COMMANDE_MOTEUR_AG=AG_COMMANDEE_G
[DEMANDE_CAG_D] and
not [DEMANDE_CAG_G] and
[BLOCAGE_CAG==NON_BLOQUE] and
[ENCLENCHEMENT_CAG_G==LIBERE]
/CAG=D;
COMMANDE_MOTEUR_AG=AG_COMMANDEE_D
[CONTROLE_POSITION_AG==IND]
POSITION_CAG_
INDETERMINEE
[DEMANDE_CAG_G] and
not [DEMANDE_CAG_D] and
[BLOCAGE_CAG==NON_BLOQUE] and
[ENCLENCHEMENT_CAG_D==LIBERE]
/CAG=G;
COMMANDE_MOTEUR_AG=AG_COMMANDEE_G
[DEMANDE_CAG_D] and
not [DEMANDE_CAG_G] and
[BLOCAGE_CAG==NON_BLOQUE] and
[ENCLENCHEMENT_CAG_G==LIBERE]
/CAG=D;
COMMANDE_MOTEUR_AG=AG_COMMANDEE_D
[CONTROLE_POSITION_AG==D]
CAG_A_DROITE
GRAPHE_COMMANDE_AG

Figure 8 – Graphe_commande_ag

3.2 Graph behaviour description

It is split into two “AND” subgraphs, which are independent each other. The graph below represents the locked or unlocked state of a generic switch point. This state may be commanded by the *Terminal Technique* (by an operator).

The input “DISPOSITION_TECHNIQUE” is always set to NON_PRISES. This assumption does not reduce the validity of the behaviour.

For this reason the graph is not reduced modifying terms of it, but simply not considering the external commands: DISPOSITION_TECHNIQUE==PRISES.

It is evident that some parts of the graph might not be stimulated.

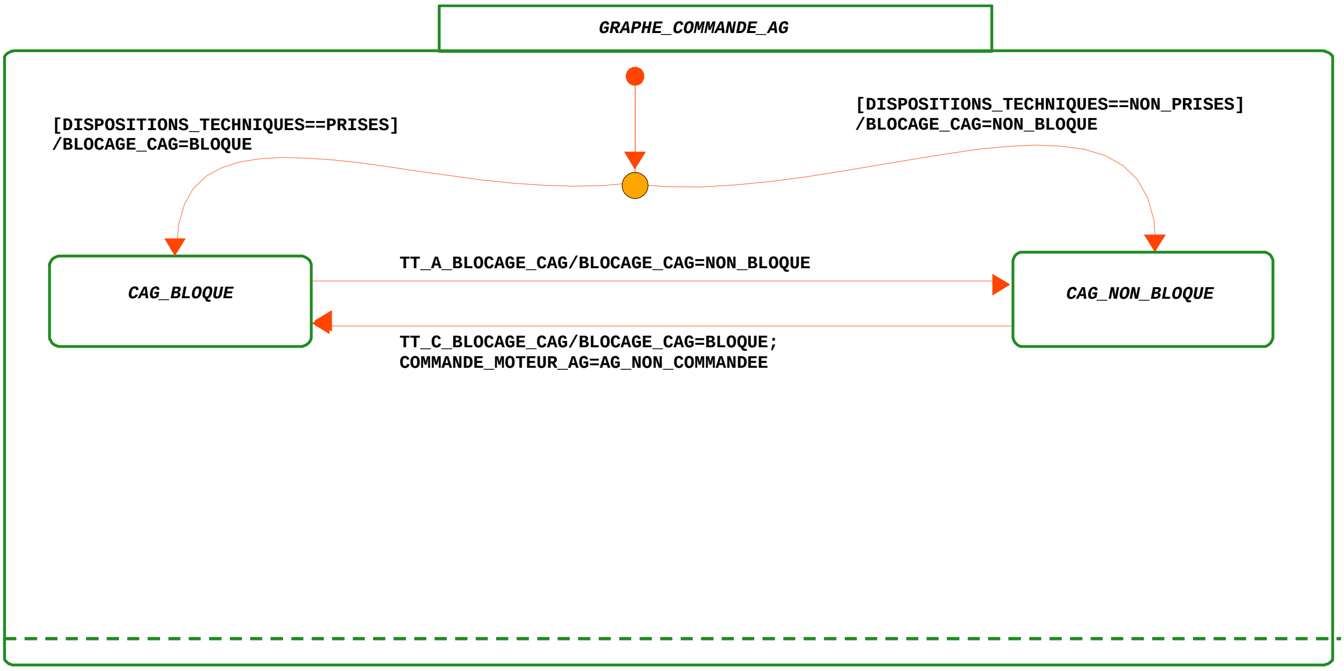

GRAPHE_COMMANDE_AG
[DISPOSITIONS_TECHNIQUES==PRISES]
/BLOCAGE_CAG=BLOQUE
[DISPOSITIONS_TECHNIQUES==NON_PRISES]
/BLOCAGE_CAG=NON_BLOQUE
CAG_BLOQUE
TT_A_BLOCAGE_CAG/BLOCAGE_CAG=NON_BLOQUE
CAG_NON_BLOQUE
TT_C_BLOCAGE_CAG/BLOCAGE_CAG=BLOQUE;
COMMANDE_MOTEUR_AG=AG_NON_COMMANDEE

Figure 9 – switch point locking

This part of the graph contains the following elements:

Command inputs	
DISPOSITIONS_TECHNIQUES	Forced to NON_PRISES
TT_A_BLOCAGE_CAG	Used
TT_C_BLOCAGE_CAG	Not used
Variables	
BLOCAGE_CAG	Used (output var)

HP: the switch point could not be locked by operators (*terminal Technique*), for this reason the TT_A_BLOCAGE_CAG may be ignored as well.

The variable is used into next graph (second part), this is only a local variable

The previous part of the graph has got only one functionality (switch locking) that we will not use.

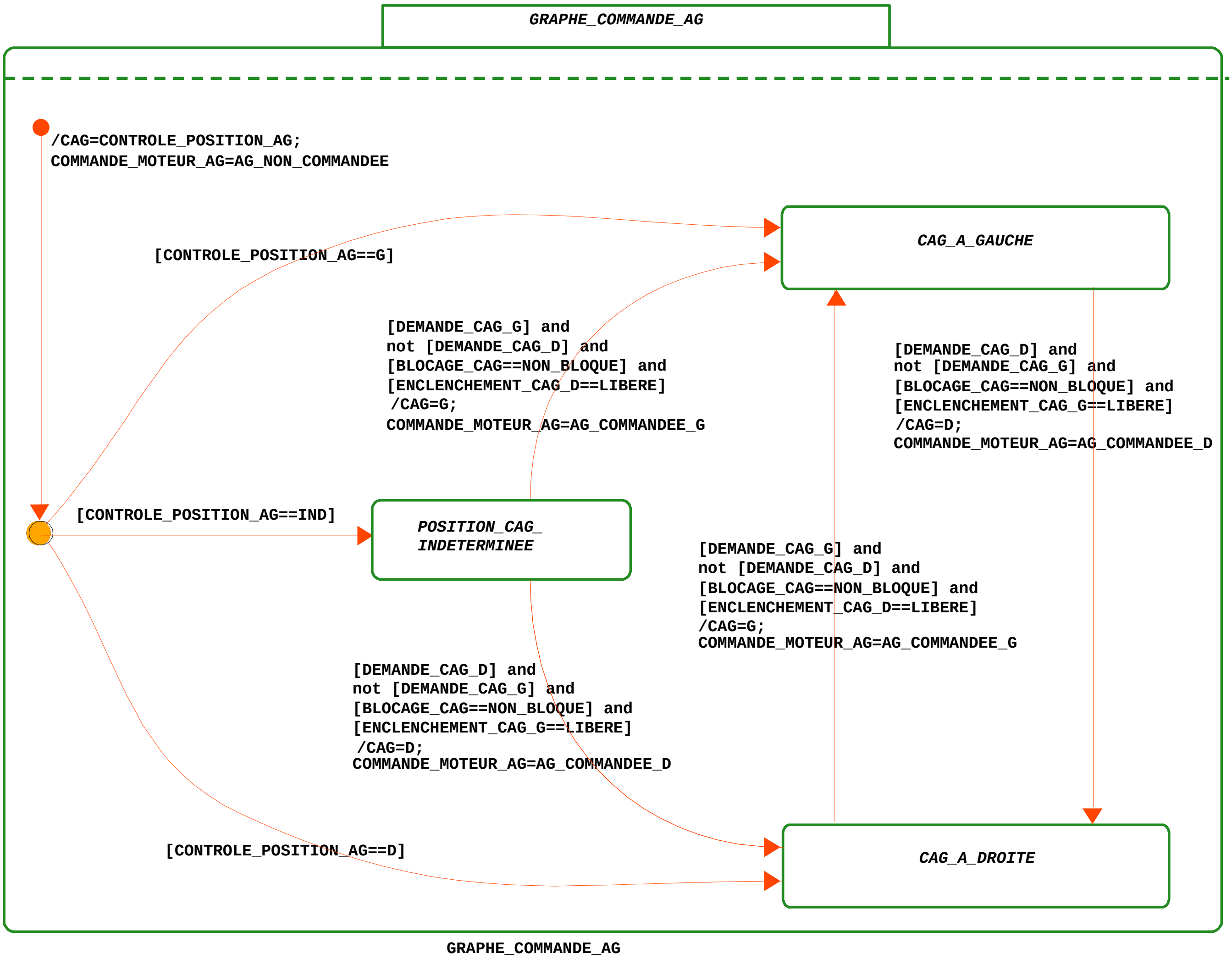

GRAPHE_COMMANDE_AG
/CAG=CONTROLE_POSITION_AG;
COMMANDE_MOTEUR_AG=AG_NON_COMMANDEE
[CONTROLE_POSITION_AG==G]
CAG_A_GAUCHE
[DEMANDE_CAG_G] and
not [DEMANDE_CAG_D] and
[BLOCAGE_CAG==NON_BLOQUE] and
[ENCLENCHEMENT_CAG_D==LIBERE]
/CAG=G;
COMMANDE_MOTEUR_AG=AG_COMMANDEE_G
[DEMANDE_CAG_D] and
not [DEMANDE_CAG_G] and
[BLOCAGE_CAG==NON_BLOQUE] and
[ENCLENCHEMENT_CAG_G==LIBERE]
/CAG=D;
COMMANDE_MOTEUR_AG=AG_COMMANDEE_D
[CONTROLE_POSITION_AG==IND]
POSITION_CAG_
INDETERMINEE
[DEMANDE_CAG_G] and
not [DEMANDE_CAG_D] and
[BLOCAGE_CAG==NON_BLOQUE] and
[ENCLENCHEMENT_CAG_D==LIBERE]
/CAG=G;
COMMANDE_MOTEUR_AG=AG_COMMANDEE_G
[DEMANDE_CAG_D] and
not [DEMANDE_CAG_G] and
[BLOCAGE_CAG==NON_BLOQUE] and
[ENCLENCHEMENT_CAG_G==LIBERE]
/CAG=D;
COMMANDE_MOTEUR_AG=AG_COMMANDEE_D
[CONTROLE_POSITION_AG==D]
CAG_A_DROITE
GRAPHE_COMMANDE_AG

Figure 10 – switch point commanded position

In Figure 10 is shown the main part of the graph:

Apart from the first initializing action depending on the physical position of the switch point when the device is initialized, the right side of the graph represents the command of position changing. Receiving a command of switch setting the graph goes from the normal position to the reverse one and viceversa.

This part of the graph contains the following elements:

Environment inputs	
CONTROLE_POSITION	Simulated, not forced, usable
ZONE	Used, it is the zone associated to the CAG

HP: the switch couldn't be commanded by operators (*terminal Technique*), for this reason the TT_DEMAND_CAG_D and TT_DEMAND_CAG_G may be ignored as well.

Output variables	
CAG	Used (output var)
COMMANDE_MOTEUR_AG	Not relevant, not used by any other graph

The main functionality of the graph consists to control the switch position: it accepts the switch position only if a set of conditions are verified.

These conditions are contained within 3 terms:

DEMAND_CAG_D/DEMAND_CAG_G
BLOCAGE_CAG
ENCLENCHEMENT_CAG_D/ ENCLENCHEMENT_CAG_G

3.2.1 Expansion Rules:

3.2.1.1 DEMAND_CAG_D (DEMAND_CAG_G)

It represents a Boolean equation that is true when a route needs to use this switch point in the normal (reverse) position.

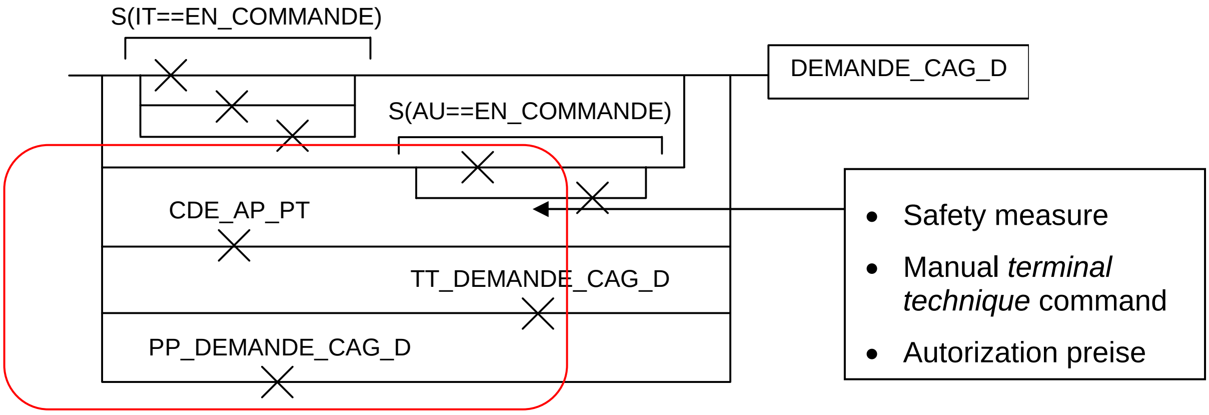


Figure 11 – Original rule

The rule has been reduced eliminating some terms.

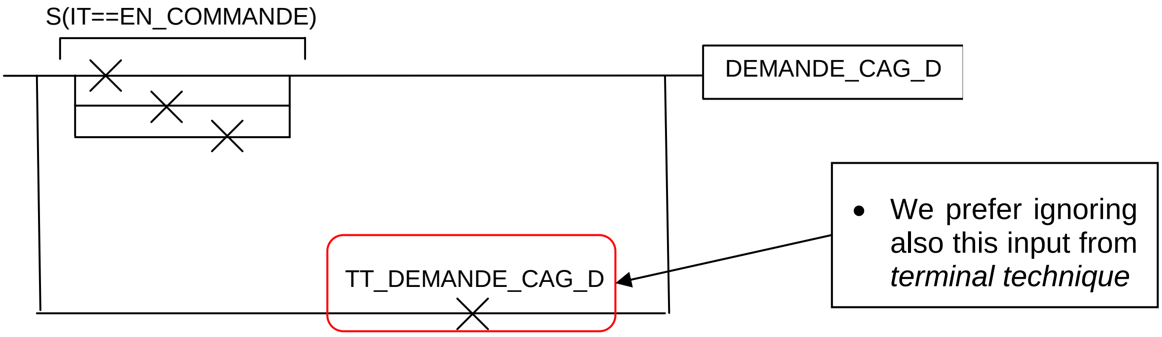

S(IT==EN_COMMANDE)
DEMANDE_CAG_D
We prefer ignoring also this input from *terminal technique*
TT_DEMANDE_CAG_D

Figure 12 – reduced rule

We consider only the variables related to routes, the other variables are dealt with in the same way, so they have been eliminated.

S(IT==EN_COMMANDE): it is the logical sum (OR) of routes which want this switch in normal (reverse) position.

These variables are not related to other graphs (no nesting).

3.2.1.2 ENCLENCHEMENT_CAG_D (ENCLENCHEMENT_CAG_D)

This rule is not completely clear.

It represents a Boolean equation that is true if the track circuit (ZONE) related to this switch is free, and furthermore there are no active routes at the same moment. (see the TRANSIT graph for more information)

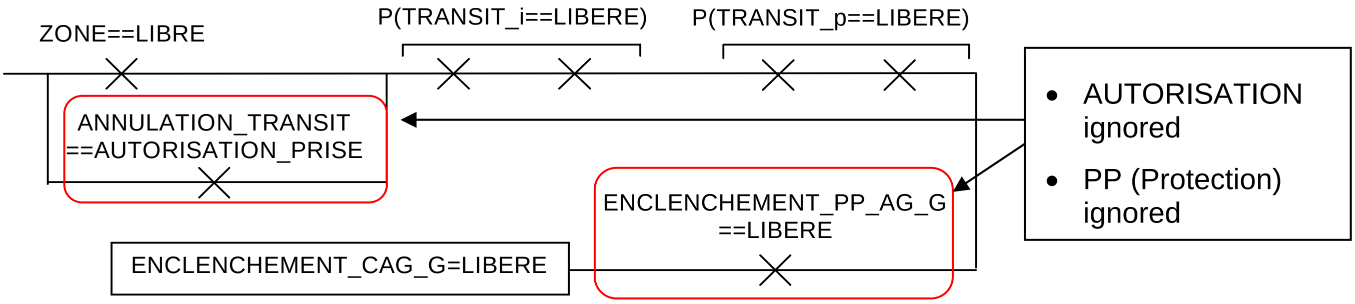


Figure 13 – original rule

The reduction of this rule is the following. The variables have been eliminated because it seems to be not clearly explained and also because they do not supply any interesting features to the expansion methodology.

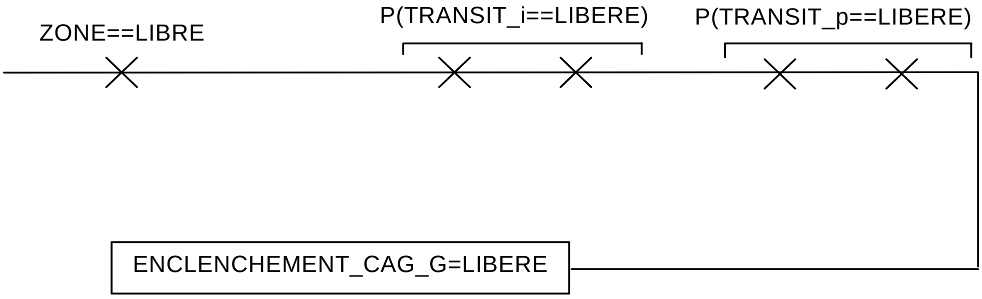


Figure 14 – reduced rule

Output to the environment:

CAG

COMMANDE_MOTEUR_AG

Variables stimulating statecharts nesting:

variable name	***Term name***	***Needed Statechart***
TRANSIT_i (impair)	ENCLENCHEMENT_CAG_D (G)	TRANSIT
TRANSIT_p (pair)	ENCLENCHEMENT_CAG_D (G)	TRANSIT

4 GRAPHE_ETAT_SIGNAL (signal state graph)

4.1 Graph description

The graph belongs to the activity chart allocated to route management.

This graph is underused related to its functionalities. In fact we have considered only the possibility to use the signal in the starting position of a route.

We have discarded the use of “intermediare signal” and “enjambe signal”.

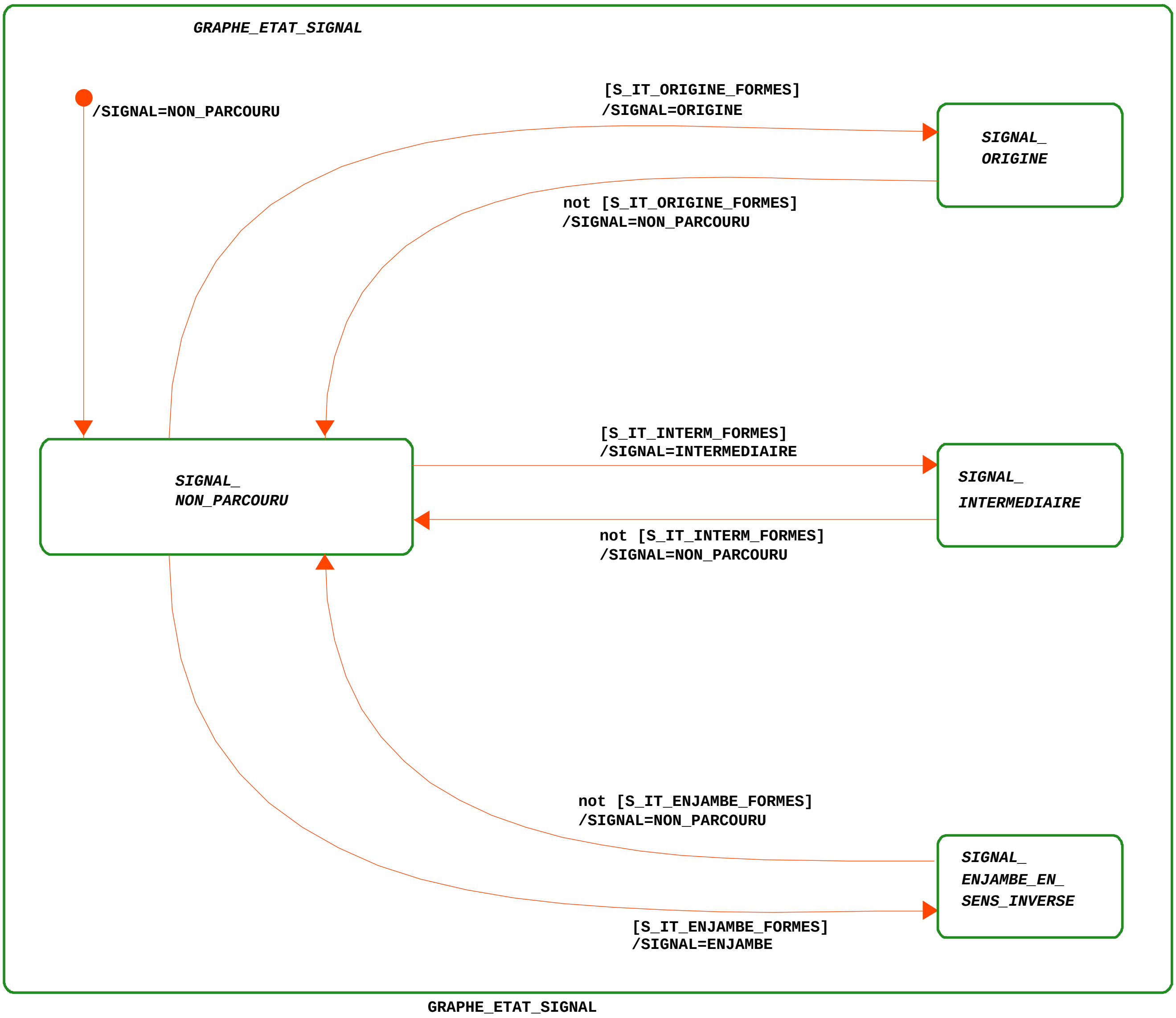


Figure 15 – Graphe_etat_signal

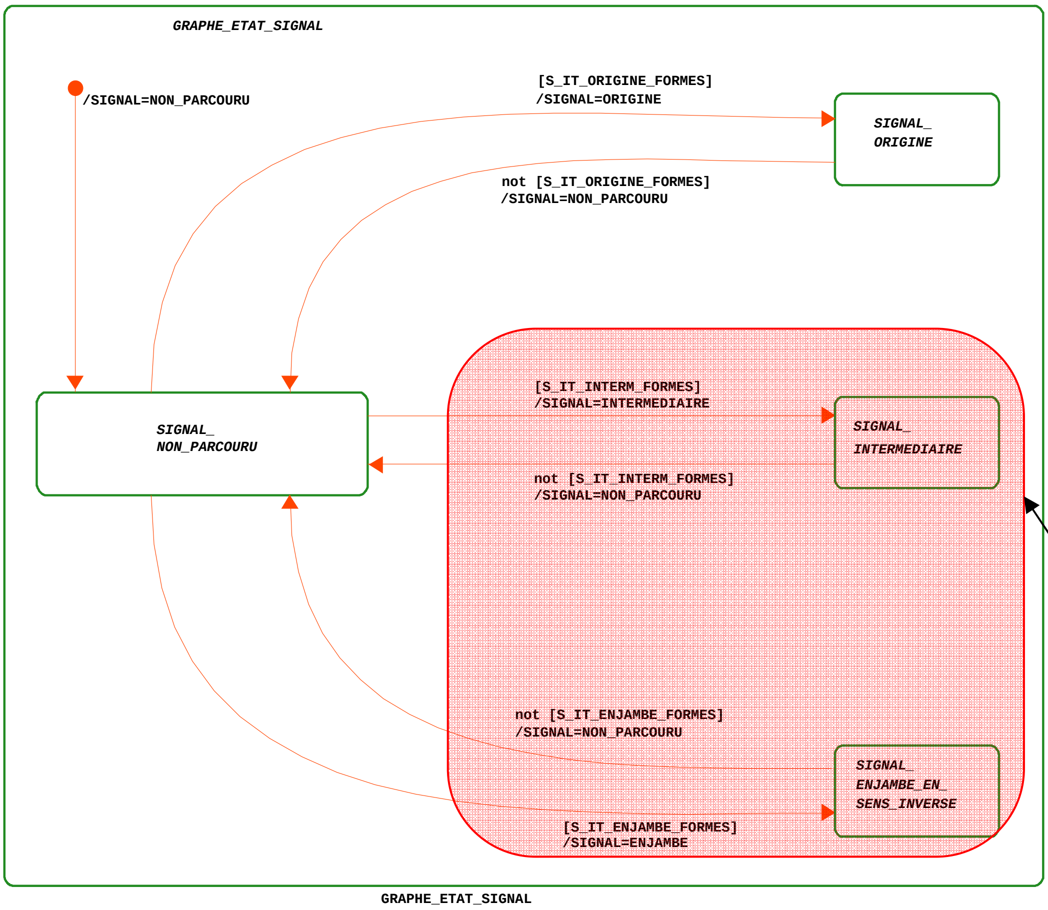
4.2 Graph behaviour description

The graph above represents the state of use of a signal. This state may be commanded by the routes.

We have considered only the part of the graph related to the starting position of the state of a signal. This assumption not reduce the validity of the behaviour (this is a functional reduction)

For this reason the graph is not reduced modifying terms, but simply not considering the terms: S_IT_ENJAMBE_FORMES, S_IT_INTERM_FORMES.

It is evident that some parts of the graph might not be used.


GRAPHE_ETAT_SIGNAL
/SIGNAL=NON_PARCOURU
[S_IT_ORIGINE_FORMES] /SIGNAL=ORIGINE
SIGNAL_ ORIGINE
not [S_IT_ORIGINE_FORMES] /SIGNAL=NON_PARCOURU
SIGNAL_ NON_PARCOURU
[S_IT_INTERM_FORMES] /SIGNAL=INTERMEDIAIRE
SIGNAL_ INTERMEDIAIRE
not [S_IT_INTERM_FORMES] /SIGNAL=NON_PARCOURU
not [S_IT_ENJAMBE_FORMES] /SIGNAL=NON_PARCOURU
SIGNAL_ ENJAMBE_EN_ SENS_INVERSE
[S_IT_ENJAMBE_FORMES] /SIGNAL=ENJAMBE
GRAPHE_ETAT_SIGNAL

- Unused part of the graph

Figure 16 – GRAPH FUNCTIONAL REDUCTION

Terms	
S_IT_ORIGINE_FORMES	Used
S_IT_ENJAMBE_FORMES	Not Used
S_IT_INTERM_FORMES	Not used

HP: the signal could be only a starting route signal

Output variables	
SIGNAL	Used (out)

This variable is used within the first level graph (ETAT_IT)

4.2.1 *Expansion Rules*

4.2.1.1 S_IT_ORIGINE_FORMES

It represents a Boolean equation that is true if there is a route which uses this signal as a starting point signal.

The rule has not been reduced.

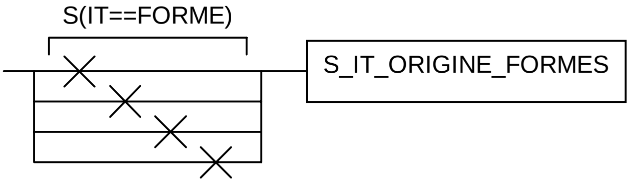


Figure 17 – Original rule

We consider only the variables related to route setting.

S(IT==FORME): it is the logical sum (OR) of routes using the signal as a starting point

These variables are related to other graphs, but no further nested.

EXAMPLES:

Tous les signaux représentés ici étant origines d'itinéraires :

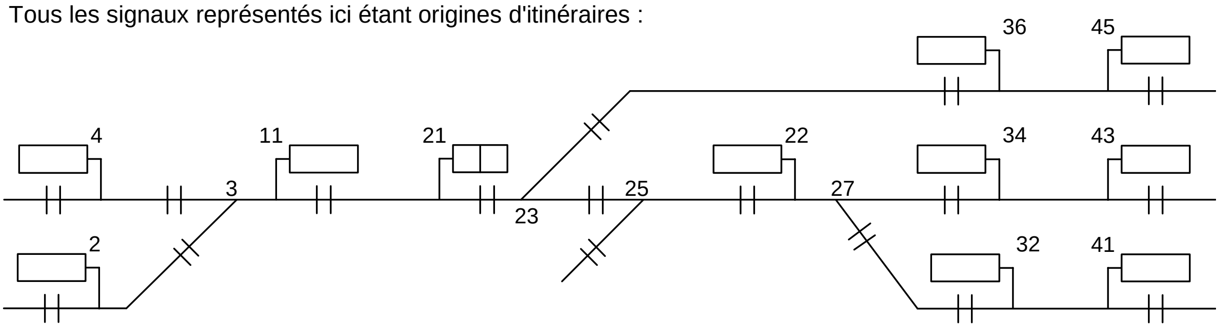
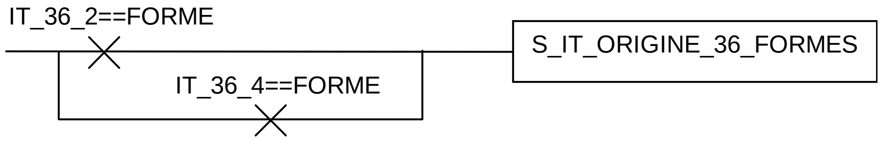


Figure 18 – yard layout

This rule has not been reduced.

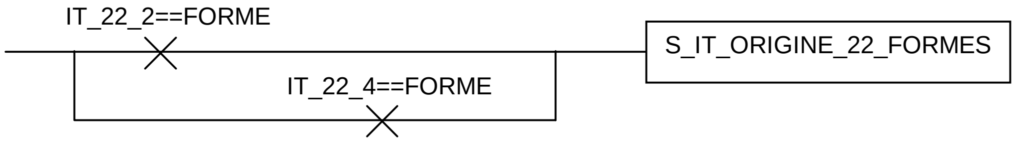
Signal 36 : This signal is only a starting point signal (this information coming from the control table, from signalling principles) :

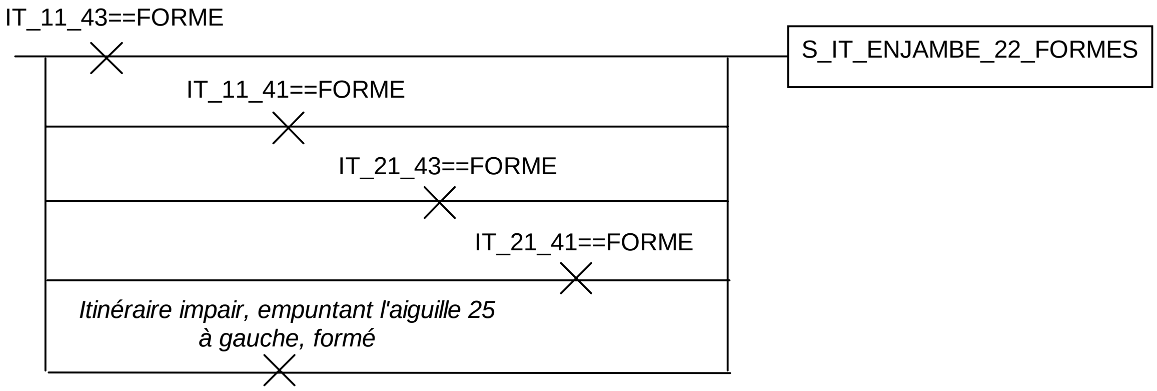


If the route from 36 to 2 is in the FORME state, then the signal 36 is set to the ORIGINE state.

The terms *S_IT_INTERM_36_FORMES* and *S_IT_ENJAMBE_36_FORMES* are always FALSE.

Signal 22 : This signal does not be anytime an intermediate signal :





The S_IT_ENJAMBE_22_FORME is a not considered functionality of the graph.

In order to command the route 11_45 the state of the signals 36 and 21 has to be evaluated.

Whether the state of one of these signals is ORIGINE, it means that there is another already set route.

Output to the environment:

SIGNAL

Variables stimulating statecharts nesting:

None.

5 GRAPHE_DA_IT_AVEC_DPG (automatic route destruction with wheel detector DPG)

5.1 Graph description

In this graph we have done a functional reduction as well, in fact we have not considered the case of no destructible route (permanent route).

The graph belongs to the activity chart allocated to route management.

Nesting: no graphs will be nested by this graph.

This graph is need because the variable DESTR_AUTO_IT is needed in the first graph in order to destruct the route.

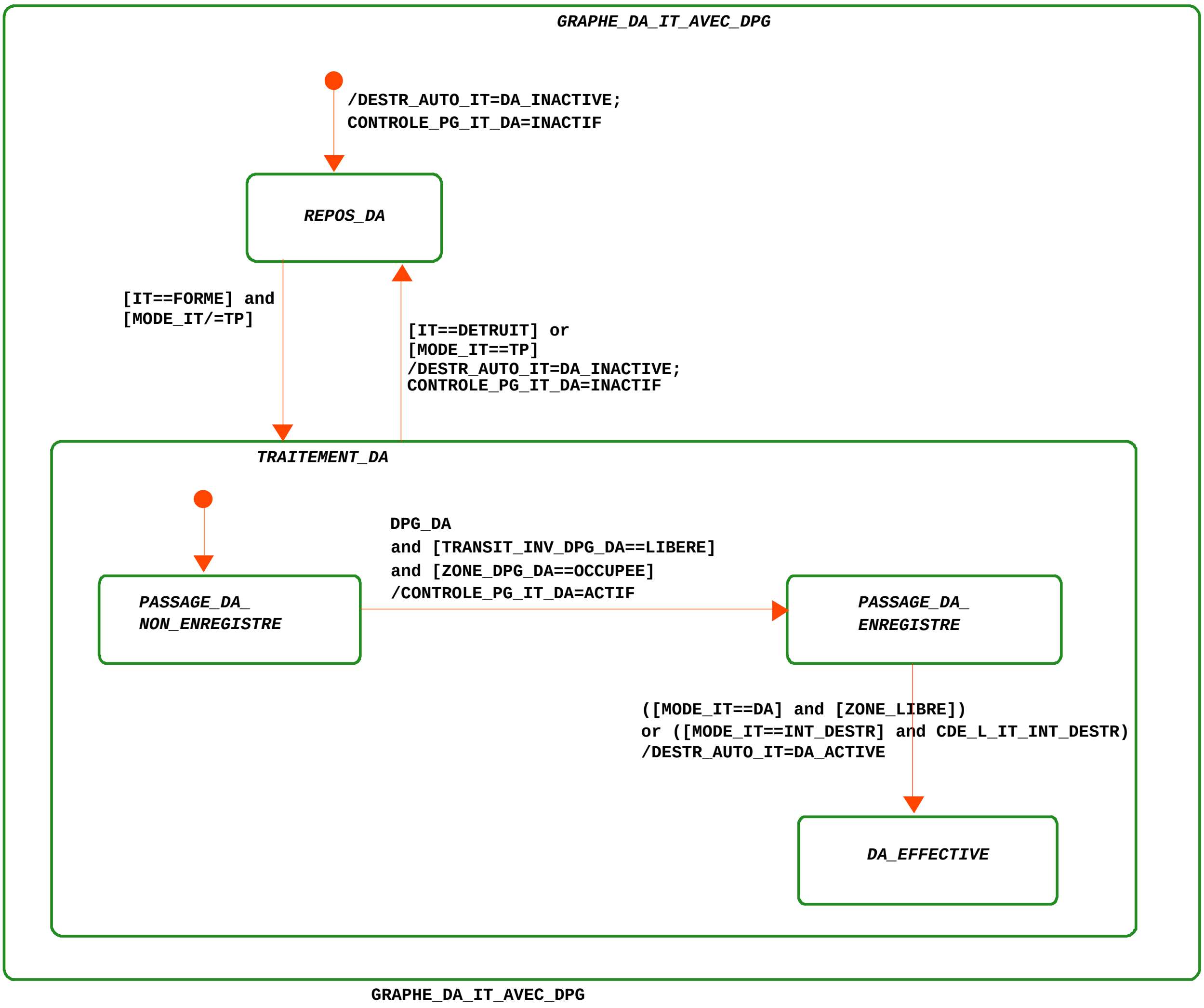

GRAPHE_DA_IT_AVEC_DPG
/DESTR_AUTO_IT=DA_INACTIVE;
CONTROLE_PG_IT_DA=INACTIF
REPOS_DA
[IT==FORME] and
[MODE_IT/=TP]
[IT==DETRUIT] or
[MODE_IT==TP]
/DESTR_AUTO_IT=DA_INACTIVE;
CONTROLE_PG_IT_DA=INACTIF
TRAITEMENT_DA
DPG_DA
and [TRANSIT_INV_DPG_DA==LIBERE]
and [ZONE_DPG_DA==OCCUPEE]
/CONTROLE_PG_IT_DA=ACTIF
PASSAGE_DA_
NON_ENREGISTRE
PASSAGE_DA_
ENREGISTRE
([MODE_IT==DA] and [ZONE_LIBRE])
or ([MODE_IT==INT_DESTR] and CDE_L_IT_INT_DESTR)
/DESTR_AUTO_IT=DA_ACTIVE
DA_EFFECTIVE
GRAPHE_DA_IT_AVEC_DPG

Figure 19 – Graphe_da_it_avec_dpg

5.2 Graph behaviour description

It is constituted by two big states: REPOS_DA and TRAITEMENT_DA

When one route is formed and the modality of route command is like automatic destruction, then this graph may go to TRAITEMENT_DA state.

Inside this state the system has to perform a sequence in order to destruct the route when a wheel detector is activated by trains.

There is a transition state, the system arrives in this state when a wheel detector is activated but the zone related to it is not completely free yet.

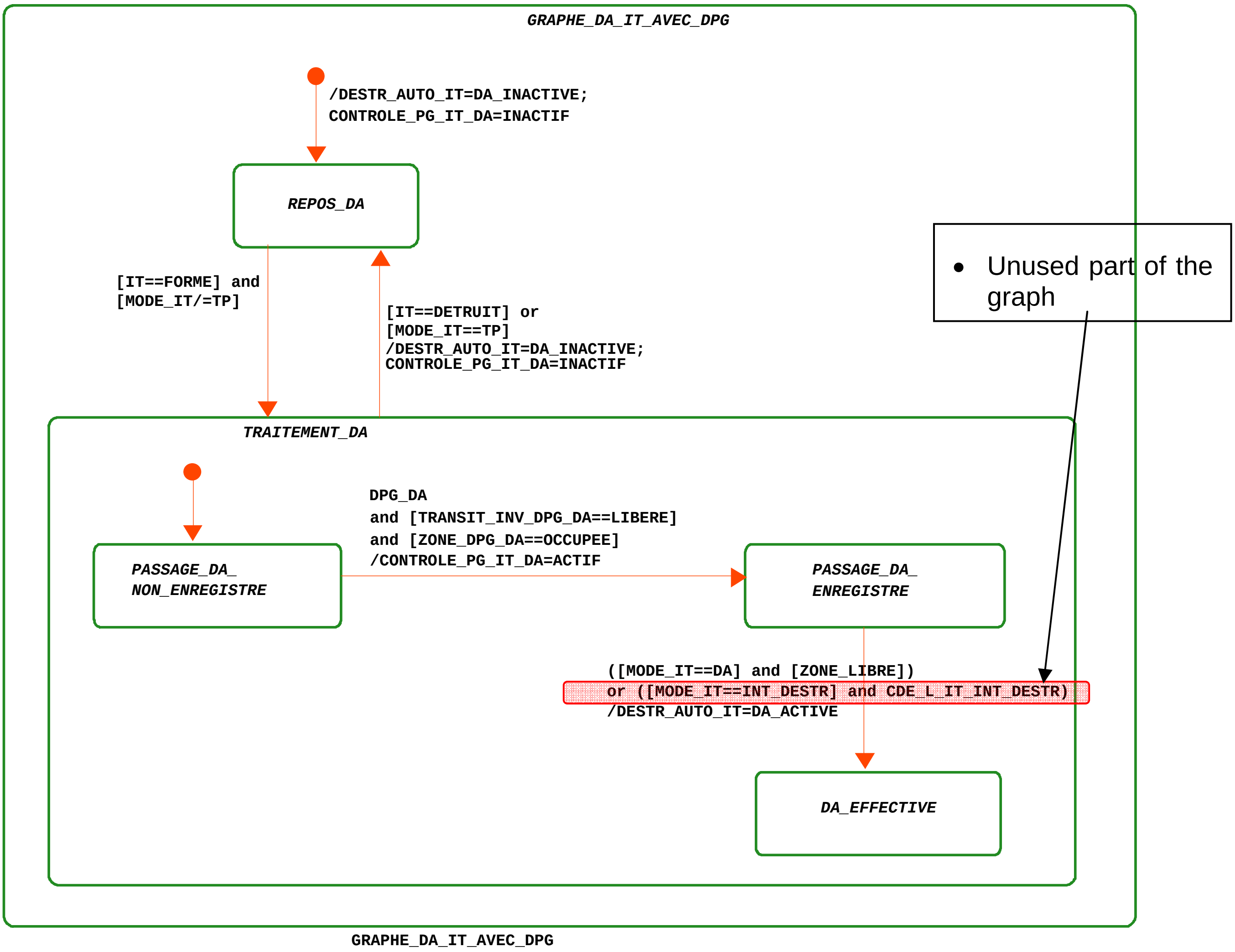

GRAPHE_DA_IT_AVEC_DPG
/DESTR_AUTO_IT=DA_INACTIVE;
CONTROLE_PG_IT_DA=INACTIF
REPOS_DA
[IT==FORME] and
[MODE_IT/=TP]
[IT==DETRUIT] or
[MODE_IT==TP]
/DESTR_AUTO_IT=DA_INACTIVE;
CONTROLE_PG_IT_DA=INACTIF
• Unused part of the graph
TRAITEMENT_DA
DPG_DA
and [TRANSIT_INV_DPG_DA==LIBERE]
and [ZONE_DPG_DA==OCCUPEE]
/CONTROLE_PG_IT_DA=ACTIF
PASSAGE_DA_
NON_ENREGISTRE
PASSAGE_DA_
ENREGISTRE
([MODE_IT==DA] and [ZONE_LIBRE])
or ([MODE_IT==INT_DESTR] and CDE_L_IT_INT_DESTR)
/DESTR_AUTO_IT=DA_ACTIVE
DA_EFFECTIVE
GRAPHE_DA_IT_AVEC_DPG

Figure 20 – unused part of the graph

We have done the functional reduction about the kind of route command (see first level).

Command inputs	
DPG_DA	Used
ZONE_DPG_DA	Used

Variables	
DESTR_AUTO_IT	Used (out)
CONTROLE_PG_IT_DA	Not used (out)
IT	Used (in)

MODE_IT	Used (in)

5.2.1 *Expansion Rules*

5.2.1.1 TRANSIT_INV_DPG_DA

Term always considered “LIBERE” (intrinsic orientation of the detector).

5.2.1.2 ZONE_LIBRE

It represents a Boolean equation that is true if the zone related to the wheel detector is free. It means that the train has crossed the detector.

The rule has not been reduced.

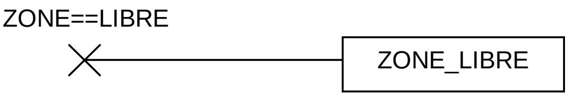


Figure 21 – Original rule

Output to the environment:

DESTR_AUTO_IT

Variables stimulating statecharts nesting:

NONE

THIRD LEVEL:

GRAPHE_TRANSIT

6 GRAPHE_TRANSIT

6.1 Graph description

The graph belongs to the activity chart allocated to route management.

We have reached this graph because of the variable TRANSIT, following the terms ENCLENCHEMENT_CAG_D, ENCLENCHEMENT_CAG_G of the GRAPHE_COMMANDE_CAG.

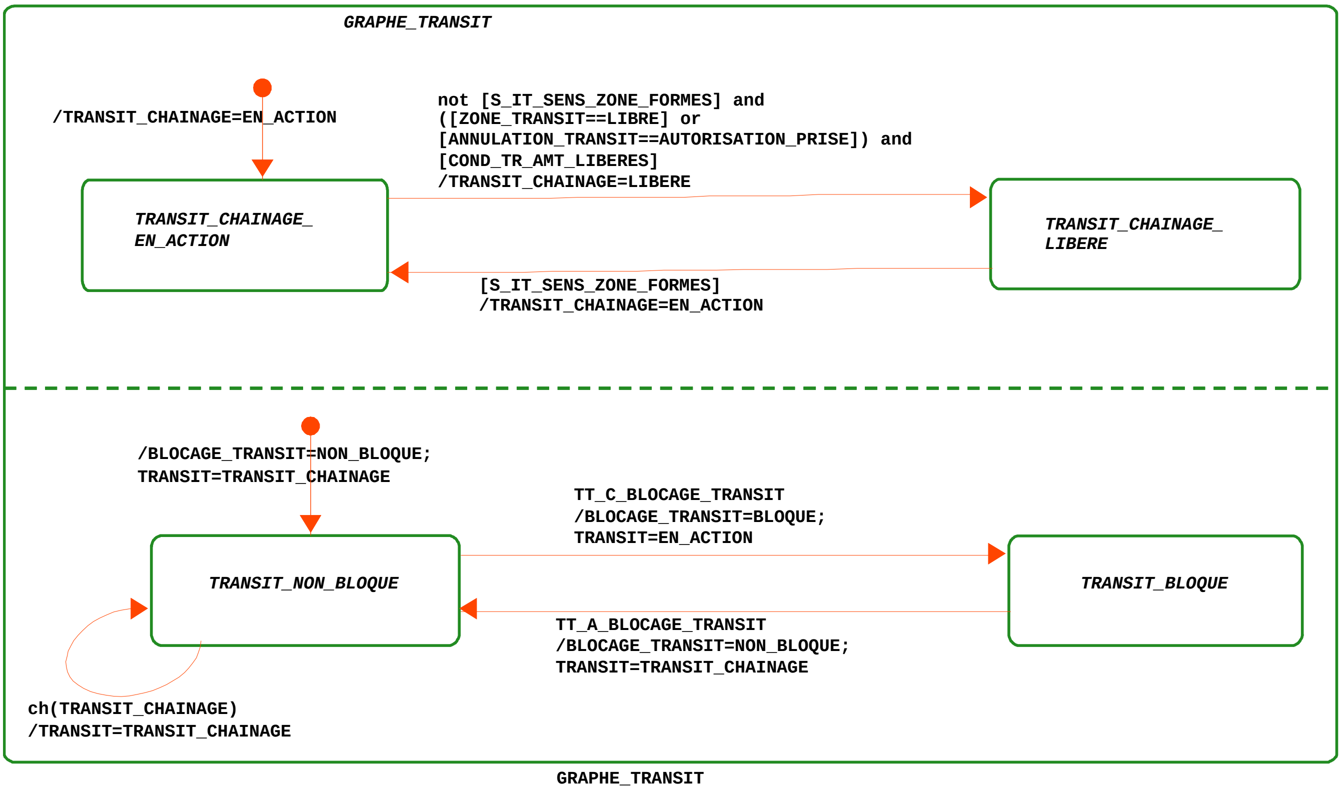


Figure 22 – Graphe_transit

6.2 Graph behaviour description

It is split into two AND subgraph, which are independent each other.

It is composed by two subgraphs, one related to the control of a chain of transit and the other to lock the transit by operator using the *Terminal Technique*.

The graph below represents the state locked or unlocked of transit. This state may be commanded by the *Terminal Technique* (by an operator).

The input "TT_C_BLOCAGE_TRANSIT" (event) is always set to false. This assumption not reduce the validity of the behaviour.

For this reason the graph is not reduced modifying its terms, but simply not considering the external commands: TT_C_BLOCAGE_TRANSIT .

It is evident that some parts of the graph might not be used.

The transit is always possible.

- Main action of this part of the graph.

- Unused state

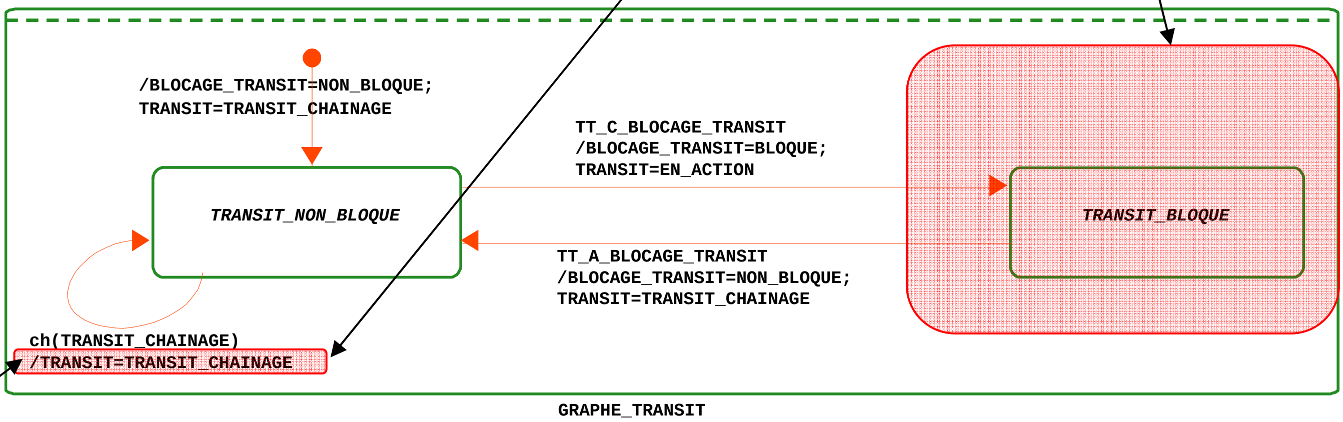


Figure 23 – Transit locking

- Ch(x) = Changed(x)
Which occurs when and if there was a change in the value of the data item expression **x**

Command inputs	
TT_C_BLOCAGE_TRANSIT	Not used
TT_A_BLOCAGE_TRANSIT	Not used

HP: the transit couldn't be locked by operators (*terminal Technique*)

Variables	
BLOCAGE_TRANSIT	Not used (out)
TRANSIT	Used (out) [nesting variable]
TRANSIT_CHAINAGE	Used within the graph

And referred to the main AND-part of the graph

- Undefined term rule

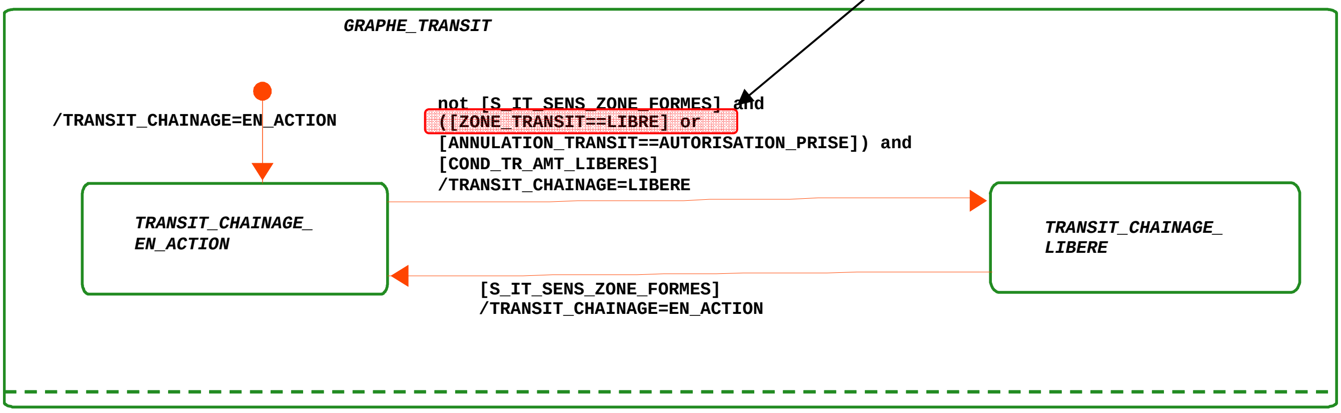

GRAPHE_TRANSIT
/TRANSIT_CHAINAGE=EN_ACTION
not [S_IT_SENS_ZONE_FORMES] and
([ZONE_TRANSIT==LIBRE] or
[ANNULATION_TRANSIT==AUTORISATION_PRISE]) and
[COND_TR_AMT_LIBERES]
/TRANSIT_CHAINAGE=LIBERE
TRANSIT_CHAINAGE_
EN_ACTION
TRANSIT_CHAINAGE_
LIBERE
[S_IT_SENS_ZONE_FORMES]
/TRANSIT_CHAINAGE=EN_ACTION

Figure 24 – main part

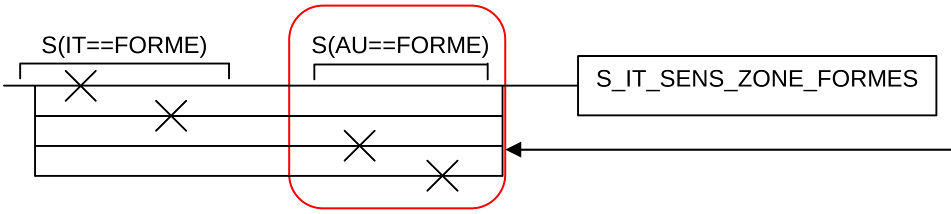
In Figure 24 is shown the main part of the graph:

The graph represents the state of a transit chain. A transit chain is a set of zones related to a route.

Command inputs	
ZONE	Used (track circuit)

6.2.1 *Expansion Rules*

6.2.1.1 S_IT_SENS_ZONE_FORMES



- Autorization: not used

Figure 25 – Original rule

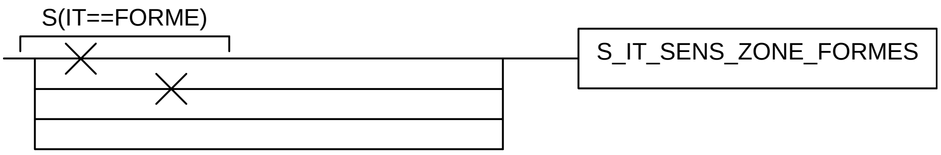


Figure 26 – Reduced rule

6.2.1.2 COND_TR_AMT_LIBERES

It represents a Boolean equation that is related to the upstream transit.

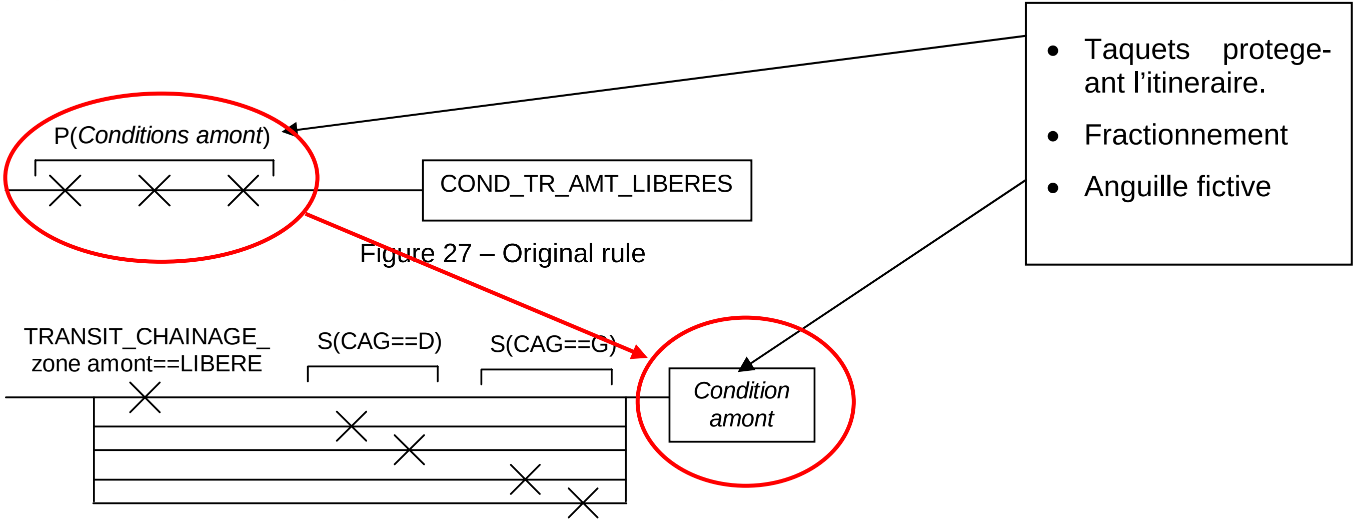
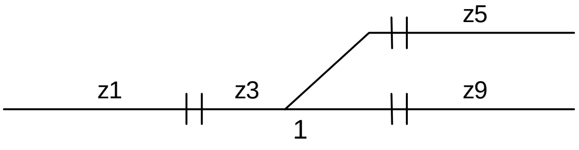


Figure 27 – Original rule

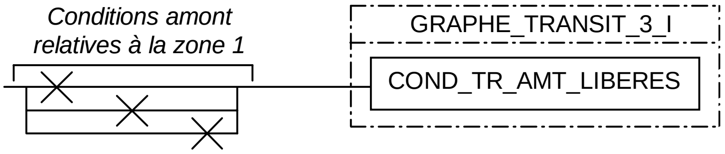
Figure 28 – original rule

"CONDITIONS AMONT": it is defined for the term COND_TR_AMT_LIBERES, it resume the combination of switch points positions of the associated zone to the transit crossed (associée au transit prises en talon) :

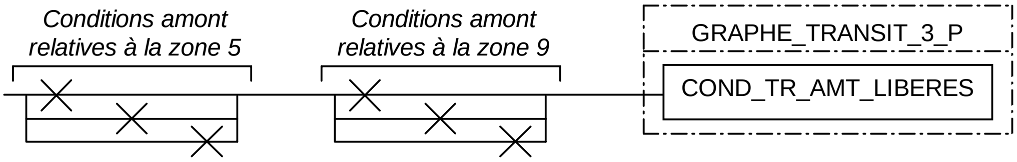
Example :



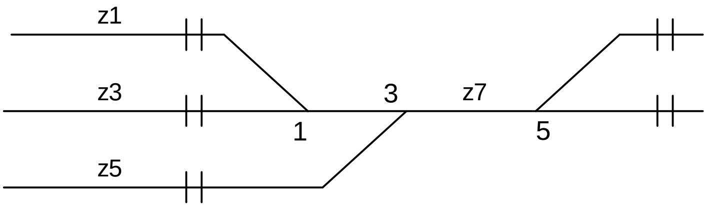
Sur la zone 3 en sens impair, il n'y a pas d'aiguille prise en talon. La combinaison d'aiguilles associée aux aiguilles prises en talon se réduit au minimum : les conditions amont du transit 3 impair se réduisent à celles relatives à la zone 1 :



Sur la zone 3 en sens pair, l'aiguille 1 est prise en talon.
A la combinaison d'aiguilles {CAG 1 à Gauche} se rapportent les conditions amont relatives à la zone 5 ;
A la combinaison d'aiguilles {CAG 1 à Droite} se rapportent les conditions amont relatives à la zone 9 :



Autre exemple :

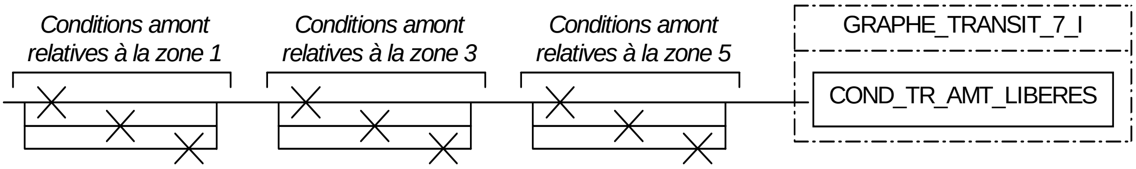


Sur la zone 7 en sens impair, les aiguilles 1 et 3 sont prises en talon.

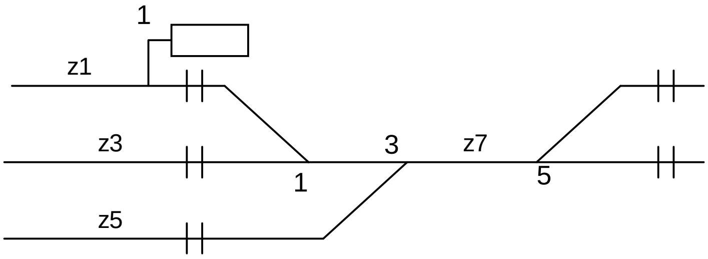
A la combinaison d'aiguilles {CAG 1 à Droite, CAG 3 à Droite} se rapportent les conditions amont relatives à la zone 1 ;

A la combinaison d'aiguilles {CAG 1 à Gauche, CAG 3 à Droite} se rapportent les conditions amont relatives à la zone 3 ;

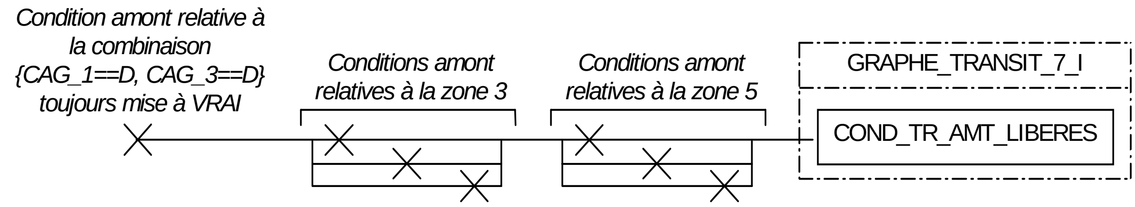
A la combinaison d'aiguilles {CAG 3 à Gauche} se rapportent les conditions amont relatives à la zone 5 :



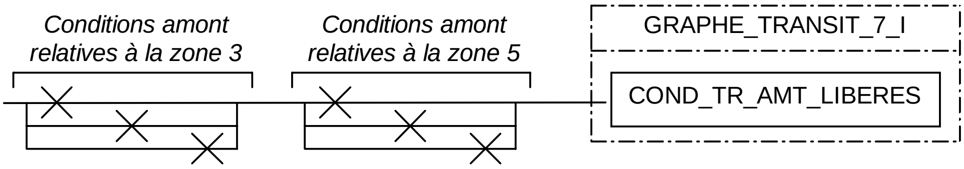
Variante à l'exemple précédent :



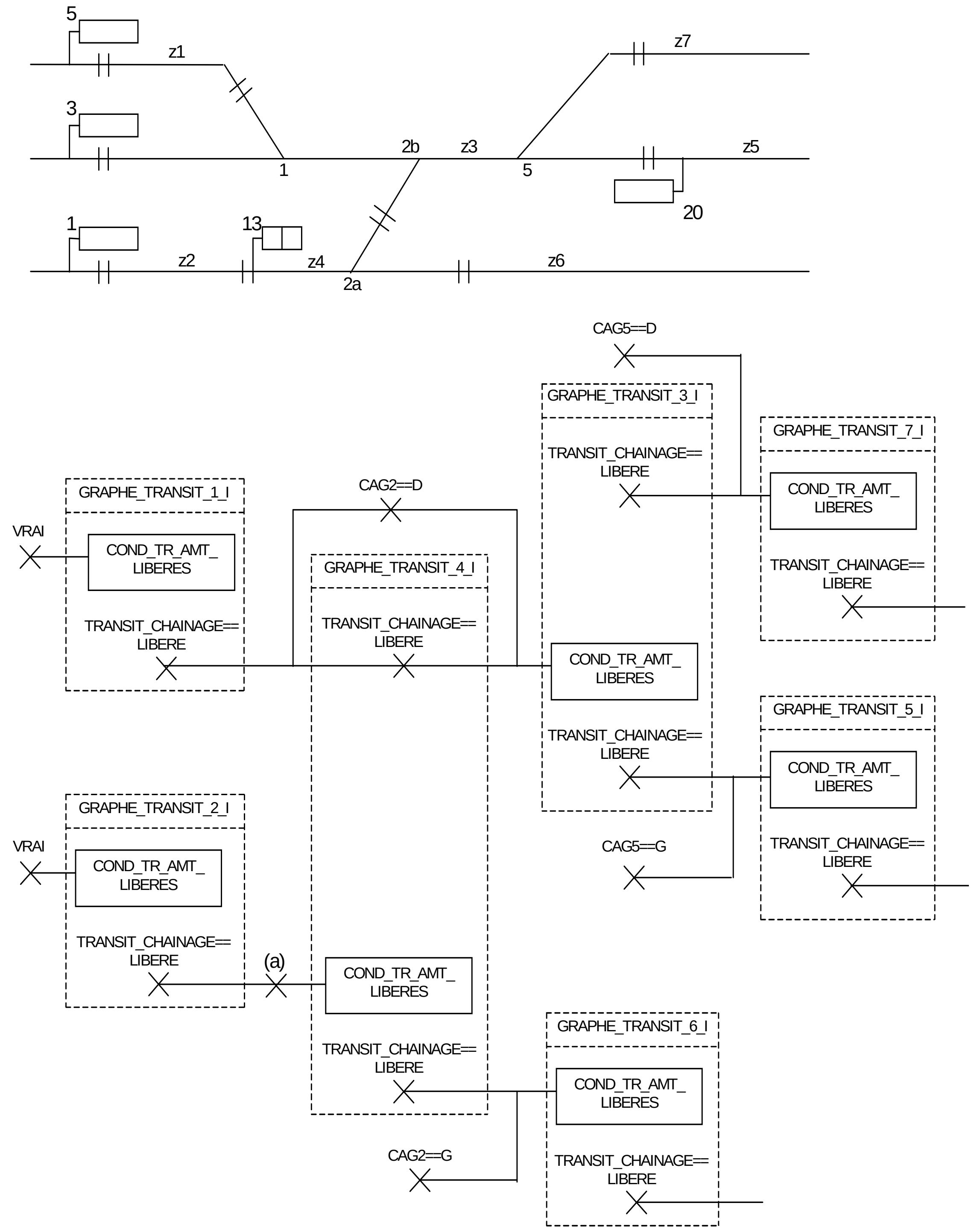
Le signal 1 étant origine d'itinéraires (et n'étant jamais signal intermédiaire), la condition amont relative à la combinaison d'aiguilles {CAG 1 à Droite, CAG 3 à Droite} est toujours mise à VRAI :


Condition amont relative à la combinaison {CAG_1==D, CAG_3==D} toujours mise à VRAI
Conditions amont relatives à la zone 3
Conditions amont relatives à la zone 5
GRAPHE_TRANSIT_7_I
COND_TR_AMT_LIBERES

Ce qui équivaut à :


Conditions amont relatives à la zone 3
Conditions amont relatives à la zone 5
GRAPHE_TRANSIT_7_I
COND_TR_AMT_LIBERES

EXEMPLE DE CONSTITUTION D'UNE CHAINE DE TRANSITS :



(a) *Condition* BANALISATION_INTERMEDIAIRE_13==EN_ACTION.
Cette condition est inutile si les commandes de blocage des transits sont neutralisées en sécurité en dehors des essais du poste.

6.2.1.3 ZONE_TRANSIT

Undefined rule

Which is the meaning of this term? (“Zone de circuit de voie associée au transit”)

7 Conclusions

After the reduction, we obtain a system still able to perform a route formation and a route destruction.

NESTED GRAPHS

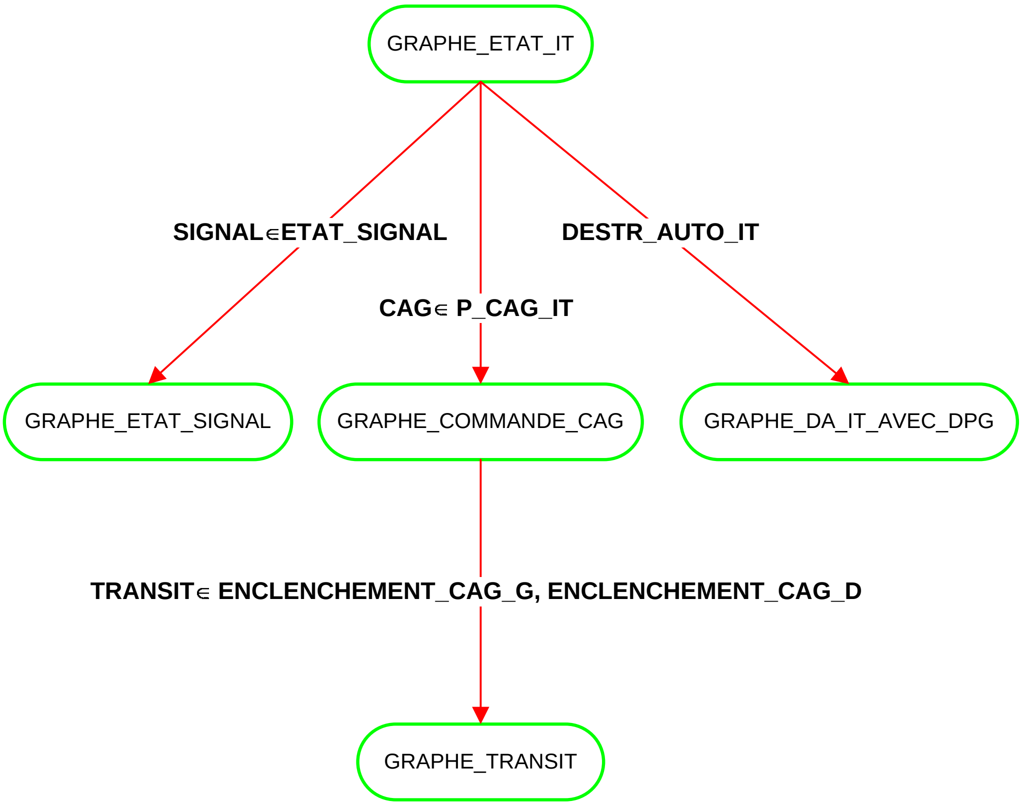


Figure 29 – Structure of graph nesting leading the graphs exploration