SNCF Model Reduction

Identification of a subset of SNCF statecharts and Rules
# Revisions

<table>
<thead>
<tr>
<th>Revision</th>
<th>Data</th>
<th>Nome</th>
<th>Pages</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>18 December 2004</td>
<td>42</td>
<td>Preliminary version</td>
</tr>
</tbody>
</table>

**Revision** 1  
**File Name** SNCF model reduction.doc

<table>
<thead>
<tr>
<th>Name</th>
<th>Signature</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Written</td>
<td>Michele Banci</td>
<td>18 December 2004</td>
</tr>
<tr>
<td>Approved</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Index

1 INTRODUCTION 6
  1.1 Definitions 6
  1.2 Purpose of the document 6

2 GRAPHE_ETAT_IT (ROUTE STATE GRAPH) 8
  2.1 Graph description 8
  2.2 Graph behaviour description 10
    2.2.1 Expansion Rules 11
      2.2.1.1 COMPL_DESTR_IT 11
      2.2.1.2 P_CAG_IT 12
      2.2.1.3 COMPL_FORMATION_IT (Complementary conditions for route setting) 12

3 GRAPHE_COMMANDE_AG (SWITCH POINT COMMANDER GRAPH) 16
  3.1 Graph description 16
  3.2 Graph behaviour description 18
    3.2.1 Expansion Rules: 20
      3.2.1.1 DEMAND_CAG_D (DEMAND_CAG_G) 20
      3.2.1.2 ENCLENCEMENT_CAG_D (ENCLENCEMENT_CAG_D) 21

4 GRAPHE_ETAT_SIGNAL (SIGNAL STATE GRAPH) 23
  4.1 Graph description 23
  4.2 Graph behaviour description 24
    4.2.1 Expansion Rules 26
      4.2.1.1 S_IT_ORIGINE_FORMES 26

5 GRAPHE_DA_IT_AVEC_DPG (AUTOMATIC ROUTE DESTRUCTION WITH WHEEL DETECTOR DPG) 28
  5.1 Graph description 28
  5.2 Graph behaviour description 29
    5.2.1 Expansion Rules 31
      5.2.1.1 TRANSIT_INV_DPG_DA 31
      5.2.1.2 ZONE_LIBRE 31

6 GRAPHE_TRANSIT 33
6.1 Graph description  33
6.2 Graph behaviour description  34
6.2.1 Expansion Rules  35
  6.2.1.1 S_IT_SENS_ZONE_FORMES  35
  6.2.1.2 COND_TR_AMT_LIBERES  36
  6.2.1.3 ZONE_TRANSIT  40

7 CONCLUSIONS  41
INDEX OF FIGURES

Figure 1 – Graphe_etat_it ........................................................................................................9
Figure 2 – choice of the kind of route .....................................................................................10
Figure 3 – route formation and destruction ............................................................................11
Figure 4 – Original rule .......................................................................................................12
Figure 5 – reduced rule ........................................................................................................12
Figure 6 – original rule ........................................................................................................13
Figure 7 – reduced rule ........................................................................................................13
Figure 8 – Graphe_commande_ag .......................................................................................17
Figure 9 – switch point locking .............................................................................................18
Figure 10 – switch point commanded position ......................................................................19
Figure 11 – Original rule .....................................................................................................21
Figure 12 – reduced rule .....................................................................................................21
Figure 13 – original rule .....................................................................................................22
Figure 14 – reduced rule .....................................................................................................22
Figure 15 – Graphe_etat_signal .........................................................................................24
Figure 16 – GRAPH FUNCTIONAL REDUCTION .................................................................25
Figure 17 – Original rule ...................................................................................................26
Figure 18 – yard layout .......................................................................................................26
Figure 19 – Graphe_da_it_avec_dpg ...................................................................................29
Figure 20 – unused part of the graph ...................................................................................30
Figure 21 – Original rule ...................................................................................................31
Figure 22 – Graphe_transit ..................................................................................................33
Figure 23 – Transit locking .................................................................................................34
Figure 24 – main part .........................................................................................................35
Figure 25 – Original rule ...................................................................................................36
Figure 26 – Reduced rule ..................................................................................................36
Figure 27 – Original rule ...................................................................................................36
Figure 28 – original rule ...................................................................................................36
Figure 29 – Structure of graph nesting leading the graphs exploration ..................................42
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 perform 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).
Command inputs

<table>
<thead>
<tr>
<th>Command</th>
<th>Used/Not Used</th>
</tr>
</thead>
<tbody>
<tr>
<td>CMD_IT_DA</td>
<td>used</td>
</tr>
<tr>
<td>CMD_IT_TP</td>
<td>not used</td>
</tr>
<tr>
<td>CMD_IT_INT_DESTR</td>
<td>not used</td>
</tr>
</tbody>
</table>

Variables

<table>
<thead>
<tr>
<th>Variable</th>
<th>Used</th>
</tr>
</thead>
<tbody>
<tr>
<td>IT</td>
<td>used</td>
</tr>
</tbody>
</table>

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

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.

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.
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.

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) :
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:**

<table>
<thead>
<tr>
<th>Variable name</th>
<th>Term name</th>
<th>Needed Statechart</th>
</tr>
</thead>
<tbody>
<tr>
<td>CAG</td>
<td>P_CAG_IT</td>
<td>COMMANDE_AG</td>
</tr>
<tr>
<td>SIGNAL</td>
<td>COMPLFORMATION_IT</td>
<td>ETAT_SIGNAL</td>
</tr>
<tr>
<td>DESTR_AUTO_IT</td>
<td></td>
<td>DA_IT_AVEC_DPG</td>
</tr>
</tbody>
</table>
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.
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.

![Graph behaviour description](image)

Figure 9 – switch point locking

This part of the graph contains the following elements:

<table>
<thead>
<tr>
<th>Command inputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>DISPOSITIONS_TECHNIQUES</td>
</tr>
<tr>
<td>TT_A_BLOCAGE_CAG</td>
</tr>
<tr>
<td>TT_C_BLOCAGE_CAG</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>BLOCAGE_CAG</td>
</tr>
</tbody>
</table>
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.

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 vice versa.

This part of the graph contains the following elements:

### Environment inputs

<table>
<thead>
<tr>
<th>Environment inputs</th>
<th>Simulated, not forced, usable</th>
</tr>
</thead>
<tbody>
<tr>
<td>CONTROLE_POSITION</td>
<td>Simulated, not forced, usable</td>
</tr>
<tr>
<td>ZONE</td>
<td>Used, it is the zone associated to the CAG</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Output variables</th>
<th>Used (output var)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CAG</td>
<td>Used (output var)</td>
</tr>
<tr>
<td>COMMANDE_MOTEUR_AG</td>
<td>Not relevant, not used by any other graph</td>
</tr>
</tbody>
</table>

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:

<table>
<thead>
<tr>
<th>DEMAND_CAG_D/DEMAND_CAG_G</th>
</tr>
</thead>
<tbody>
<tr>
<td>BLOCAGE_CAG</td>
</tr>
<tr>
<td>ENCLENCEMENT_CAG_D/ ENCLENCEMENT_CAG_G</td>
</tr>
</tbody>
</table>

### 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.
The rule has been reduced eliminating some terms.

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 ENCLENCEMENT_CAG_D**

(ENCLENCEMENT_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)
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.

Output to the environment:

- CAG
- COMMANDE_MOTEUR_AG

Variables stimulating statecharts nesting:

<table>
<thead>
<tr>
<th>variable name</th>
<th>Term name</th>
<th>Needed Statechart</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRANSIT_i (impair)</td>
<td>ENCLENCHEMENT_CAG_D (G)</td>
<td>TRANSIT</td>
</tr>
<tr>
<td>TRANSIT_p (pair)</td>
<td>ENCLENCHEMENT_CAG_D (G)</td>
<td>TRANSIT</td>
</tr>
</tbody>
</table>

• AUTORISATION ignored
• PP (Protection) ignored
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”.
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.

Figure 16 – GRAPH FUNCTIONAL REDUCTION

<table>
<thead>
<tr>
<th>Terms</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>S_IT_ORIGINE_FORMES</td>
<td>Used</td>
</tr>
<tr>
<td>S_IT_ENJAMBE_FORMES</td>
<td>Not Used</td>
</tr>
<tr>
<td>S_IT_INTERM_FORMES</td>
<td>Not used</td>
</tr>
</tbody>
</table>

HP: the signal could be only a starting route signal

Output variables

<table>
<thead>
<tr>
<th>SIGNAL</th>
<th>Used (out)</th>
</tr>
</thead>
</table>

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](image)

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:**

![Figure 18 – yard layout](image)
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):

\[ \text{IT}_36_2 == \text{FORME} \]
\[ \times \]
\[ \text{IT}_36_4 == \text{FORME} \]
\[ \times \]
\[ \text{S_IT_ORIGINE}_36_\text{FORMES} \]

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

The terms \( \text{S_IT_INTERM}_36_\text{FORMES} \) and \( \text{S_IT_ENJAMBE}_36_\text{FORMES} \) are always FALSE.

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

\[ \text{IT}_22_2 == \text{FORME} \]
\[ \times \]
\[ \text{IT}_22_4 == \text{FORME} \]
\[ \times \]
\[ \text{S_IT_ORIGINE}_22_\text{FORMES} \]

\[ \text{IT}_11_43 == \text{FORME} \]
\[ \times \]
\[ \text{IT}_11_41 == \text{FORME} \]
\[ \times \]
\[ \text{IT}_21_43 == \text{FORME} \]
\[ \times \]
\[ \text{IT}_21_41 == \text{FORME} \]
\[ \times \]
\[ \text{S_IT_ENJAMBE}_22_\text{FORMES} \]

*Itinéraire impair, empruntant l'aiguille 25 à gauche, formé*

The \( \text{S_IT_ENJAMBE}_22_\text{FORME} \) 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 needed because the variable DESTR_AUTO_IT is needed in the first graph in order to destruct the route.
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.
We have done the functional reduction about the kind of route command (see first level).

### Command inputs

<table>
<thead>
<tr>
<th>Command inputs</th>
<th>Used/Not used</th>
</tr>
</thead>
<tbody>
<tr>
<td>DPG_DA</td>
<td>Used</td>
</tr>
<tr>
<td>ZONE_DPG_DA</td>
<td>Used</td>
</tr>
</tbody>
</table>

### Variables

<table>
<thead>
<tr>
<th>Variables</th>
<th>Used/Not used</th>
</tr>
</thead>
<tbody>
<tr>
<td>DESTR_AUTO_IT</td>
<td>Used (out)</td>
</tr>
<tr>
<td>CONTROLE_PG_IT_DA</td>
<td>Not used (out)</td>
</tr>
<tr>
<td>IT</td>
<td>Used (in)</td>
</tr>
</tbody>
</table>
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.

ZONE==LIBRE

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.
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.

---

**Command inputs**

<table>
<thead>
<tr>
<th>Command inputs</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>TT_C_BLOCAGE_TRANSIT</td>
<td>Not used</td>
</tr>
<tr>
<td>TT_A_BLOCAGE_TRANSIT</td>
<td>Not used</td>
</tr>
</tbody>
</table>

HP: the transit couldn’t be locked by operators (terminal Technique)
Variables

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BLOCAGE_TRANSIT</td>
<td>Not used (out)</td>
</tr>
<tr>
<td>TRANSIT</td>
<td>Used (out) [nesting variable]</td>
</tr>
<tr>
<td>TRANSIT_CHAINAGE</td>
<td>Used within the graph</td>
</tr>
</tbody>
</table>

And referred to the main AND-part of the graph

- Undefined term rule

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

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZONE</td>
<td>Used (track circuit)</td>
</tr>
</tbody>
</table>

### 6.2.1 Expansion Rules

#### 6.2.1.1 S_IT_SENS_ZONE_FORMES

- Authorization: not used
6.2.1.2 COND_TR_AMT_LIBERES

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

"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:

\[
\begin{array}{c}
\text{z1} \quad \text{z3} \quad \text{z5}
\end{array}
\]

\[
\begin{array}{c}
\text{z9}
\end{array}
\]

- Taquets protege-ant l’itinéraire.
- Fractionnement
- Anguille fictive
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 :

\[
\begin{align*}
\text{Conditions amont relatives à la zone 1} & \quad \text{GRAPHE_TRANSIT_3_I} \\
& \quad \text{COND_TR_AMT_LIBERES}
\end{align*}
\]

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 :

\[
\begin{align*}
\text{Conditions amont relatives à la zone 5} & \quad \text{GRAPHE_TRANSIT_3_P} \\
\text{Conditions amont relatives à la zone 9} & \quad \text{COND_TR_AMT_LIBERES}
\end{align*}
\]

Autre exemple :

\[
\begin{align*}
z1 & \quad \text{z3} \\
\text{1} & \quad \text{3} \\
z5 & \quad \text{z7}
\end{align*}
\]

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 :

\[
\begin{align*}
\text{Conditions amont relatives à la zone 1} & \quad \text{GRAPHE_TRANSIT_7_I} \\
\text{Conditions amont relatives à la zone 3} & \quad \text{COND_TR_AMT_LIBERES}
\end{align*}
\]

\[
\begin{align*}
\text{Conditions amont relatives à la zone 5} & \quad \text{COND_TR_AMT_LIBERES}
\end{align*}
\]
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 \(\{\text{CAG 1 à Droite, CAG 3 à Droite}\}\) est toujours mise à VRAI :

Ce qui équivaut à :
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.
Figure 29 – Structure of graph nesting leading the graphs exploration