Nantes, 2009

Modélisation des Systèmes Réactifs

Synthèse de contraintes temporisées pour une architecture d'automatisation en réseau

Étienne André, Thomas Chatain, Laurent Fribourg

Laboratoire Spécification et Vérification

LSV, ENS de Cachan & CNRS, France

Olivier De Smet, Silvain Ruel

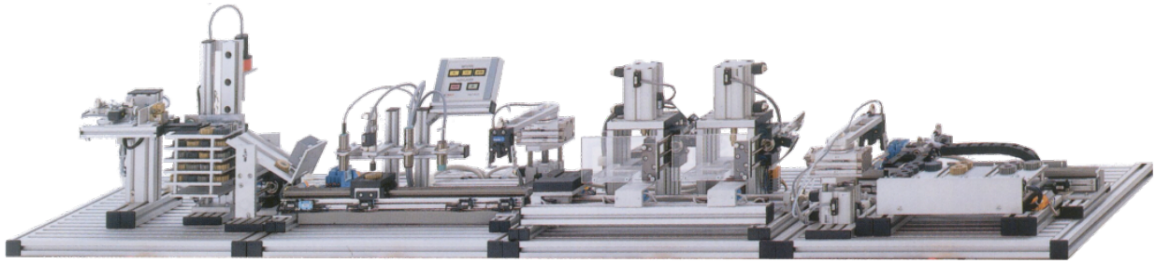
Laboratoire Universitaire de Recherche en Production Automatisée

LURPA, ENS de Cachan, Cnam, France

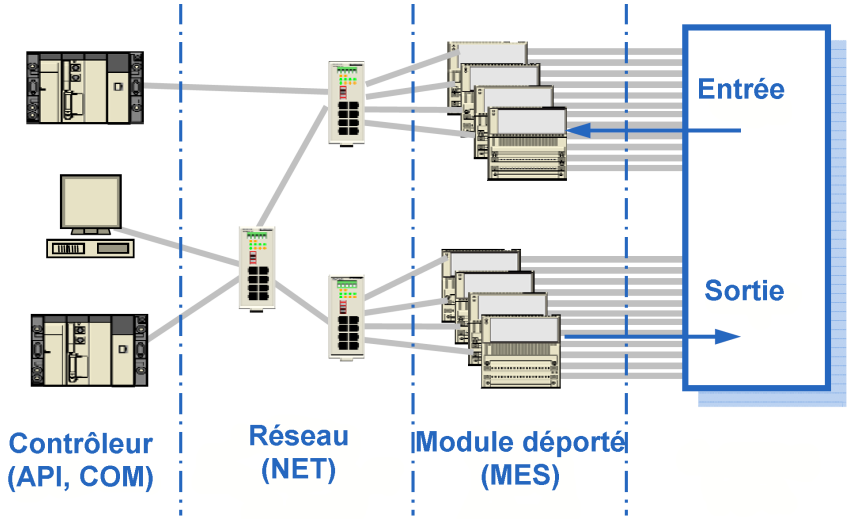
Le projet SIMOP

- SIMOP : Synergie Simulation et Model-Checking Paramétré
- Projet ENS Cachan « Farman » (2007–2009)
- Laboratoires membres
  - Laboratoire Spécification et Vérification
  - Laboratoire Universitaire de Recherche en Production Automatisée

Contexte : vérification d'un système distribué



- Architecture d'automatisation en réseau



  - Contrôleurs modulaires (API) : processeur de calcul
  - Processeur de communication (COM) : interroge le MES
  - Réseau (NET) : communication entre API et MES
  - Module déporté d'entrées–sorties (MES)

Ajustement de paramètres temporels

- Paramètres temporels du système (constantes ajustables)

  PLCct *COMct* *SIGmrt* *PLCmtt*
  RIOd *COMd* *NETd*

Ajustement de paramètres temporels

- Paramètres temporels du système (constantes ajustables)

  $PLCct$ $COMct$ $SIGmrt$ $PLCmtt$
  $RIOd$ $COMd$ $NETd$

- Détermination d'une valuation de référence des paramètres temporels

  $PLCct = 300$ $COMct = 1000$ $SIGmrt = 2071$ $PLCmtt = 100$
  $RIOd = 70$ $COMd = 25$ $NETd = 10$

  - Vérification que ces valeurs correspondent à un « bon » comportement du système (non-acessibilité des « mauvais » états)

- Question : le système sera-t-il toujours correct si l'on change ces valeurs temporelles ?

Ajustement de paramètres temporels

- Paramètres temporels du système (constantes ajustables)

  $PLCct$ $COMct$ $SIGmrt$ $PLCmtt$
  $RIOd$ $COMd$ $NETd$

- Détermination d'une valuation de référence des paramètres temporels

  $PLCct = 300$ $COMct = 1000$ $SIGmrt = 2071$ $PLCmtt = 100$
  $RIOd = 70$ $COMd = 25$ $NETd = 10$

  - Vérification que ces valeurs correspondent à un « bon » comportement du système (non-acessibilité des « mauvais » états)

- Question : le système sera-t-il toujours correct si l'on change ces valeurs temporelles ?

Objectif

Déterminer des valuations des paramètres correspondant à un bon comportement du système.

Plan

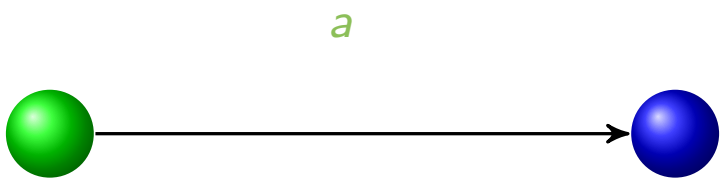
Plan

Automate temporisé

- Automate d'états fini (ensembles d'états de contrôles)

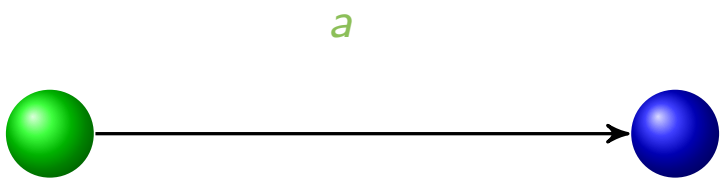
Automate temporisé

- Automate d'états fini (ensembles d'états de contrôles et d'actions)



Automate temporisé

- Automate d'états fini (ensembles d'états de contrôles et d'actions) auxquels on ajoute
  - Un ensemble X d'horloges (c.-à-d. un ensemble de variables réelles évoluant linéairement à la même vitesse) [Alur and Dill, 1994]



Automate temporisé

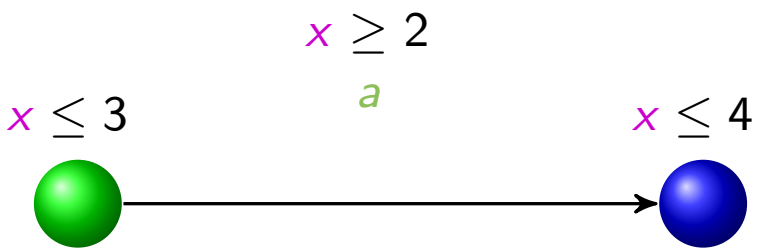
- Automate d'états fini (ensembles d'états de contrôles et d'actions) auxquels on ajoute
  - Un ensemble X d'horloges (c.-à-d. un ensemble de variables réelles évoluant linéairement à la même vitesse) [Alur and Dill, 1994]

- Caractéristiques
  - Invariant sur les états de contrôle : propriété vérifiée par les horloges pour rester dans un état de contrôle

Automate temporisé

- Automate d'états fini (ensembles d'états de contrôles et d'actions) auxquels on ajoute
  - Un ensemble X d'horloges (c.-à-d. un ensemble de variables réelles évoluant linéairement à la même vitesse) [Alur and Dill, 1994]

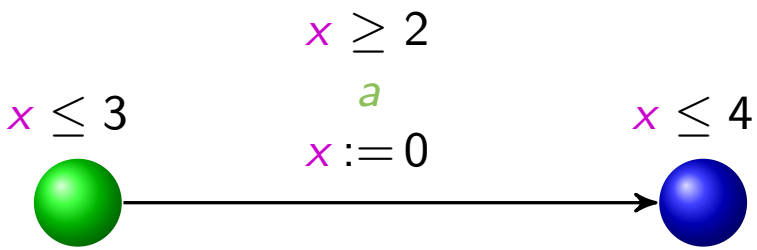
- Caractéristiques
  - Invariant sur les états de contrôle : propriété vérifiée par les horloges pour rester dans un état de contrôle
  - Gardes sur les transitions : propriété vérifiée par les horloges pour franchir une transition



Automate temporisé

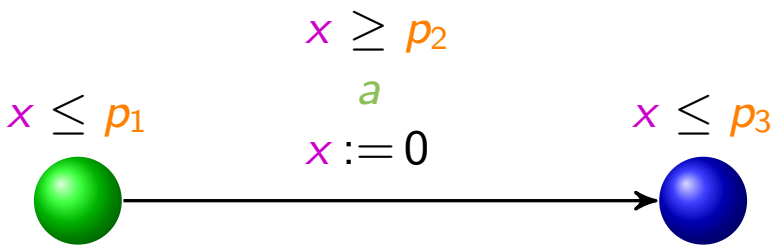
- Automate d'états fini (ensembles d'états de contrôles et d'actions) auxquels on ajoute
  - Un ensemble X d'horloges (c.-à-d. un ensemble de variables réelles évoluant linéairement à la même vitesse) [Alur and Dill, 1994]

- Caractéristiques
  - Invariant sur les états de contrôle : propriété vérifiée par les horloges pour rester dans un état de contrôle
  - Gardes sur les transitions : propriété vérifiée par les horloges pour franchir une transition
  - Réinitialisation d'horloges à 0 lors d'une transition



Automate temporisé paramétré (ATP)

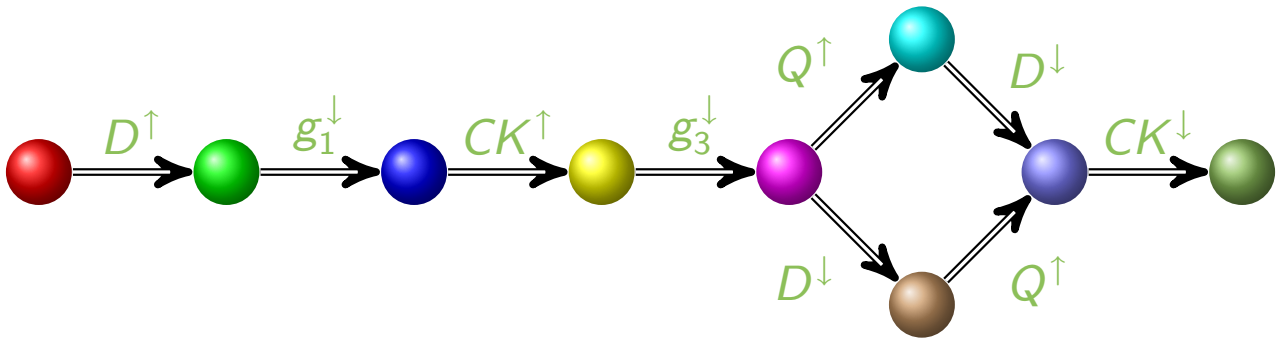
- Automate d'états fini (ensembles d'états de contrôles et d'actions) auxquels on ajoute
  - Un ensemble X d'horloges (c.-à-d. un ensemble de variables réelles évoluant linéairement à la même vitesse) [Alur and Dill, 1994]
  - Un ensemble P de paramètres (ou constantes inconnues), utilisées dans les gardes et invariants [Alur et al., 1993]
- Caractéristiques
  - Invariant sur les états de contrôle : propriété vérifiée par les horloges et les paramètres pour rester dans un état de contrôle
  - Gardes sur les transitions : propriété vérifiée par les horloges et les paramètres pour franchir une transition
  - Réinitialisation d'horloges à 0 lors d'une transition



États et traces

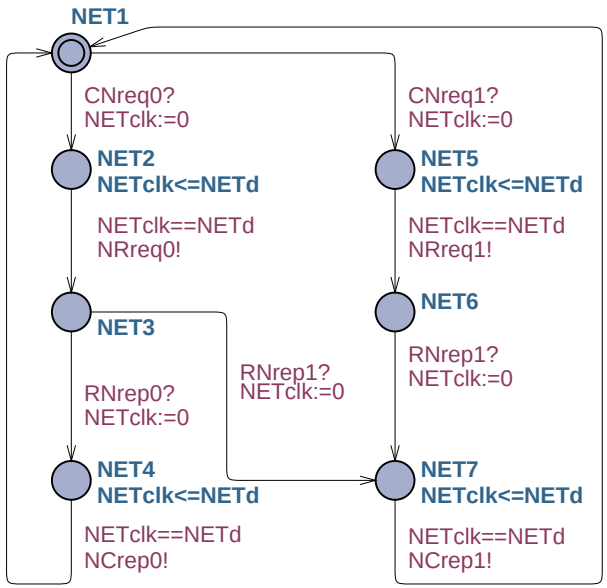
- État symbolique d'un ATP : couple (q, C), où
  - q est un état de contrôle,
  - C est une contrainte (conjonction d'inégalités) sur les paramètres

- Trace (exécution abstraction faite du temps) sur un ATP : séquence finie alternante d'états de contrôles et d'actions



Modélisation du système

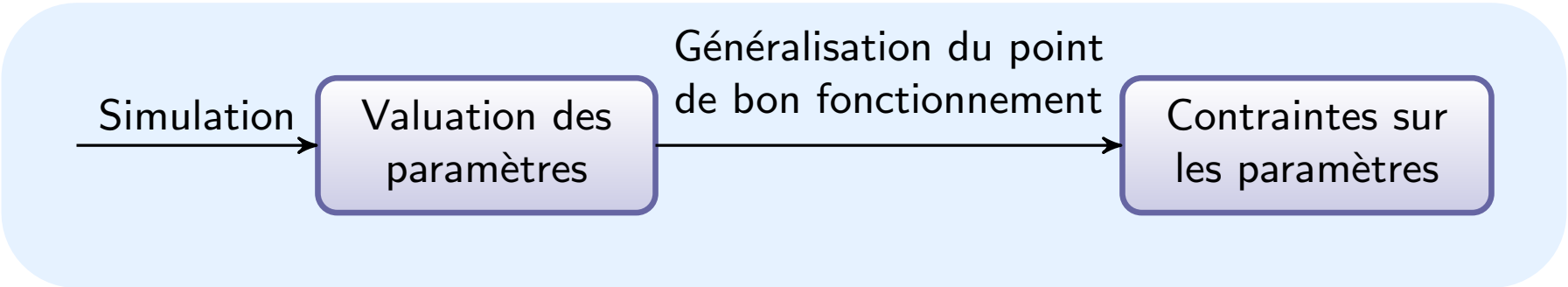
- Modélisation de chaque module du système (API, COM, NET, MES) par un automate temporisé paramétré
  - Exemple : module NET



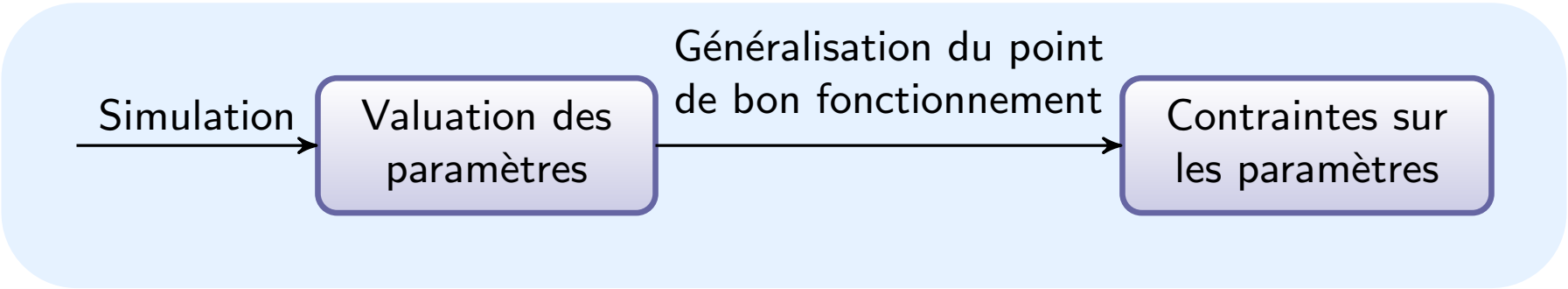
- Modélisation de l'environnement par un automate temporisé paramétré
  - Comportement des signaux d'entrée

Plan

Notre méthode



Notre méthode



Utilisation de la méthode inverse [André et al., 2009]

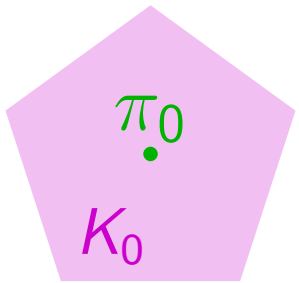
La méthode inverse

- Entrées
  - Un ATP $\mathcal{A}$
  - Une valuation de référence π_0 de tous les paramètres de $\mathcal{A}$
    - ⋆ Correspondant à un bon comportement
      (toutes les traces sous π_0 correspondent à un bon comportement)

La méthode inverse

- Entrées
  - Un ATP $\mathcal{A}$
  - Une valuation de référence π_0 de tous les paramètres de $\mathcal{A}$
    - ⋆ Correspondant à un bon comportement
      (toutes les traces sous π_0 correspondent à un bon comportement)

- Sortie : généralisation
  - Une contrainte K_0 sur les paramètres telle que
    - ⋆ $\pi_0 \models K_0$
    - ⋆ Pour toute valuation $\pi \models K_0$, l'ensemble des traces sous π est le même que l'ensemble des traces sous π_0



L'idée générale de la méthode inverse

Partir de $K_0 = True$

1. Calculer l'ensemble S des états accessibles sous K_0
2. Raffiner K_0 en supprimant un état π_0-incompatible de S
   - Sélectionner un état (q, C) π_0-incompatible dans S (c.-à-d. $\pi_0 \not\models C$)
   - Sélectionner une inégalité J π_0-incompatible dans C (c.-à-d. $\pi_0 \not\models J$)
   - Ajouter $\neg J$ à K_0
3. Aller en (1)

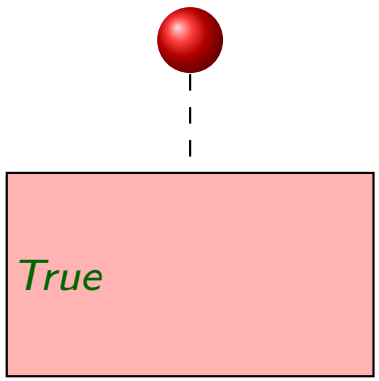
Jusqu'à point fixe (plus d'état π_0-incompatible dans S)

Application à un exemple de circuit

π_0 :

$\delta_1^- = 1$	$\delta_1^+ = 1$	$T_{HI} = 20$
$\delta_2^- = 8$	$\delta_2^+ = 10$	$T_{LO} = 15$
$\delta_3^- = 5$	$\delta_3^+ = 6$	$T_{Setup} = 10$
$\delta_4^- = 3$	$\delta_4^+ = 5$	$T_{Hold} = 15$

$K_0 = True$

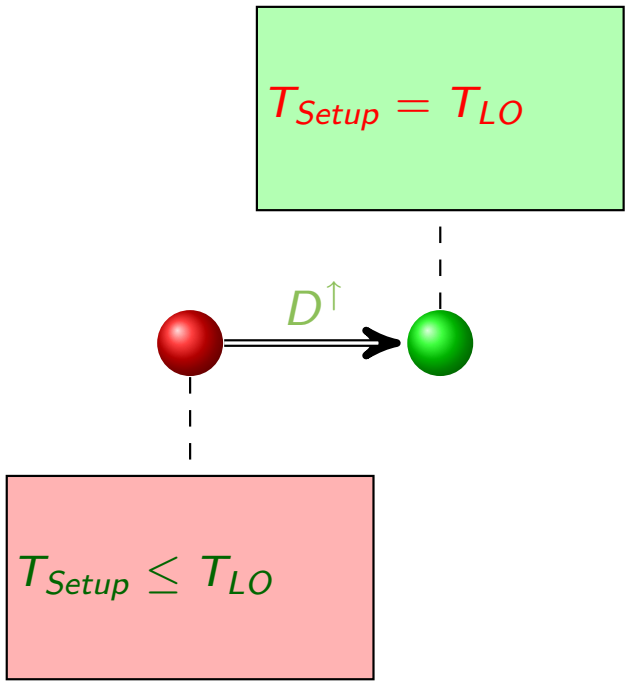


Application à un exemple de circuit

π_0 :

$\delta_1^- = 1$	$\delta_1^+ = 1$	$T_{HI} = 20$
$\delta_2^- = 8$	$\delta_2^+ = 10$	$T_{LO} = 15$
$\delta_3^- = 5$	$\delta_3^+ = 6$	$T_{Setup} = 10$
$\delta_4^- = 3$	$\delta_4^+ = 5$	$T_{Hold} = 15$

$K_0 = True$

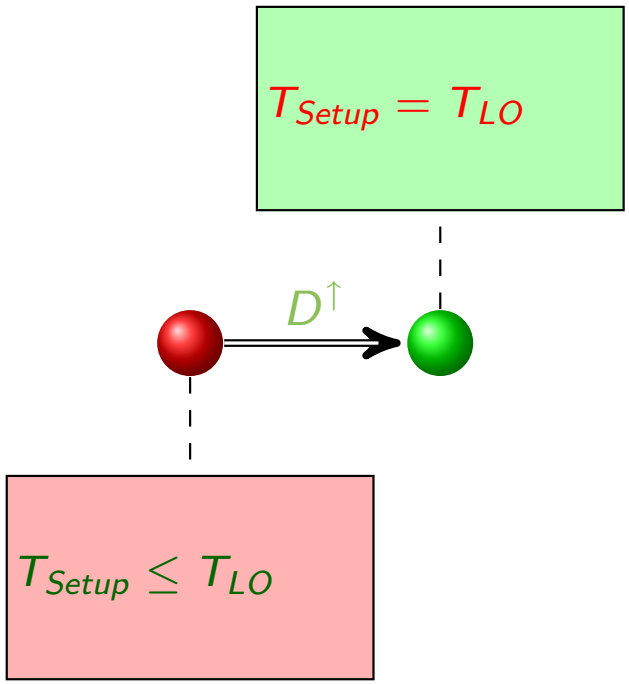


Application à un exemple de circuit

π_0 :

$\delta_1^- = 1$	$\delta_1^+ = 1$	$T_{HI} = 20$
$\delta_2^- = 8$	$\delta_2^+ = 10$	$T_{LO} = 15$
$\delta_3^- = 5$	$\delta_3^+ = 6$	$T_{Setup} = 10$
$\delta_4^- = 3$	$\delta_4^+ = 5$	$T_{Hold} = 15$

$K_0 =$
$T_{Setup} < T_{LO}$

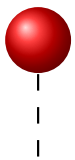


Application à un exemple de circuit

π_0 :

$\delta_1^- = 1$	$\delta_1^+ = 1$	$T_{HI} = 20$
$\delta_2^- = 8$	$\delta_2^+ = 10$	$T_{LO} = 15$
$\delta_3^- = 5$	$\delta_3^+ = 6$	$T_{Setup} = 10$
$\delta_4^- = 3$	$\delta_4^+ = 5$	$T_{Hold} = 15$

$K_0 =$
$T_{Setup} < T_{LO}$



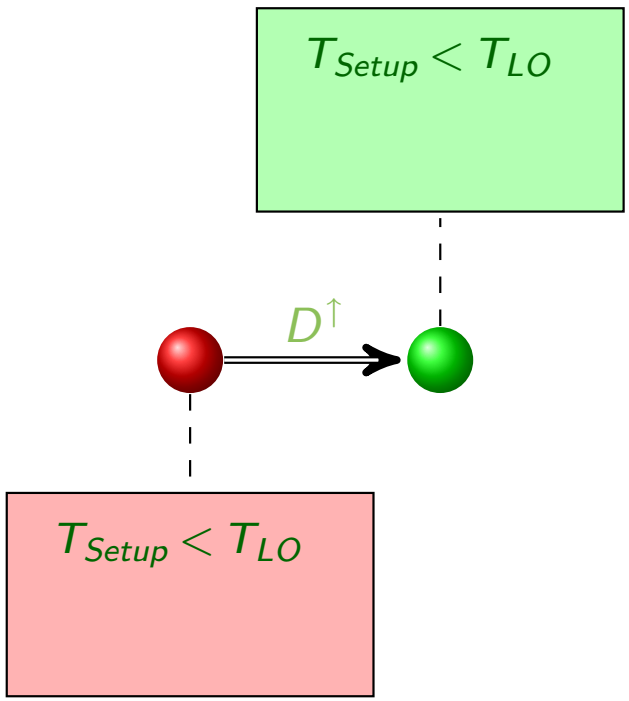
$T_{Setup} < T_{LO}$

Application à un exemple de circuit

π_0 :

$\delta_1^- = 1$	$\delta_1^+ = 1$	$T_{HI} = 20$
$\delta_2^- = 8$	$\delta_2^+ = 10$	$T_{LO} = 15$
$\delta_3^- = 5$	$\delta_3^+ = 6$	$T_{Setup} = 10$
$\delta_4^- = 3$	$\delta_4^+ = 5$	$T_{Hold} = 15$

$K_0 =$
$T_{Setup} < T_{LO}$

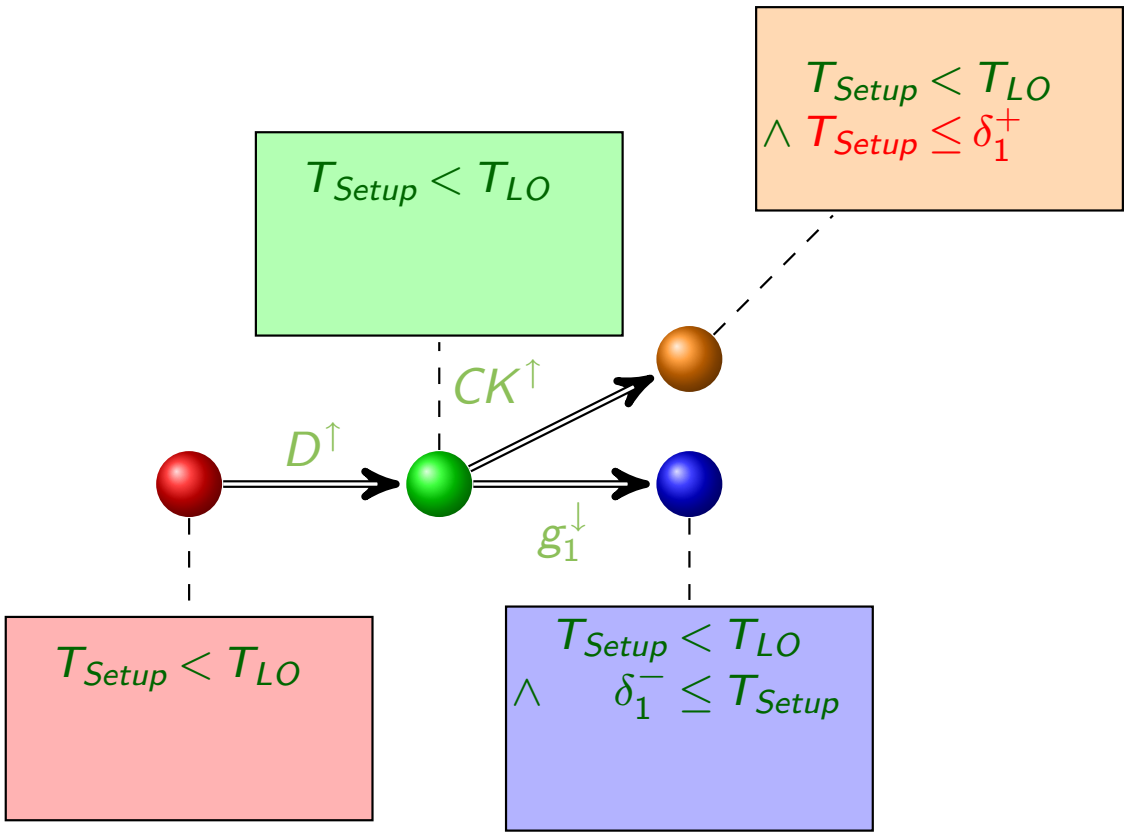


Application à un exemple de circuit

π_0 :

$\delta_1^- = 1$	$\delta_1^+ = 1$	$T_{HI} = 20$
$\delta_2^- = 8$	$\delta_2^+ = 10$	$T_{LO} = 15$
$\delta_3^- = 5$	$\delta_3^+ = 6$	$T_{Setup} = 10$
$\delta_4^- = 3$	$\delta_4^+ = 5$	$T_{Hold} = 15$

$K_0 = T_{Setup} < T_{LO}$

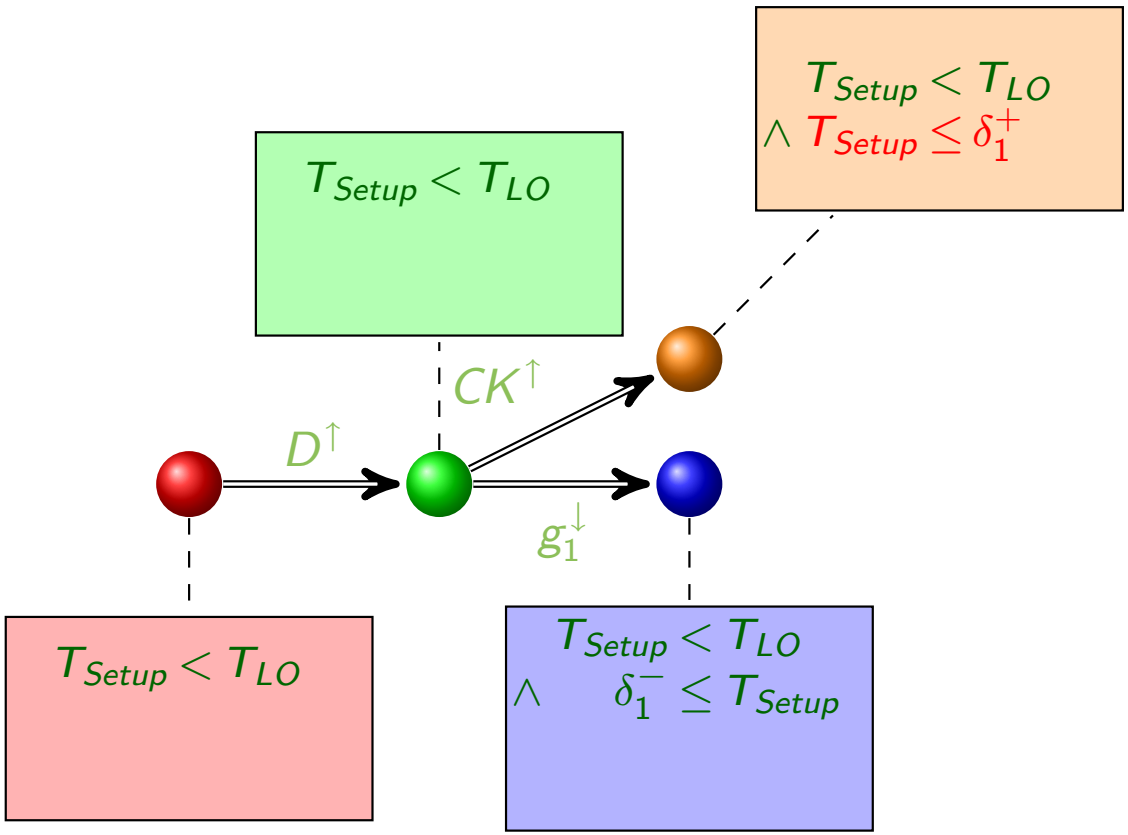


Application à un exemple de circuit

π_0 :

$\delta_1^- = 1$	$\delta_1^+ = 1$	$T_{HI} = 20$
$\delta_2^- = 8$	$\delta_2^+ = 10$	$T_{LO} = 15$
$\delta_3^- = 5$	$\delta_3^+ = 6$	$T_{Setup} = 10$
$\delta_4^- = 3$	$\delta_4^+ = 5$	$T_{Hold} = 15$

$K_0 =$
$T_{Setup} < T_{LO}$
$\wedge\ T_{Setup} > \delta_1^+$



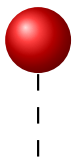
Application à un exemple de circuit

π_0 :

$\delta_1^- = 1$	$\delta_1^+ = 1$	$T_{HI} = 20$
$\delta_2^- = 8$	$\delta_2^+ = 10$	$T_{LO} = 15$
$\delta_3^- = 5$	$\delta_3^+ = 6$	$T_{Setup} = 10$
$\delta_4^- = 3$	$\delta_4^+ = 5$	$T_{Hold} = 15$

$K_0 =$

$T_{Setup} < T_{LO}$
$\wedge\ T_{Setup} > \delta_1^+$



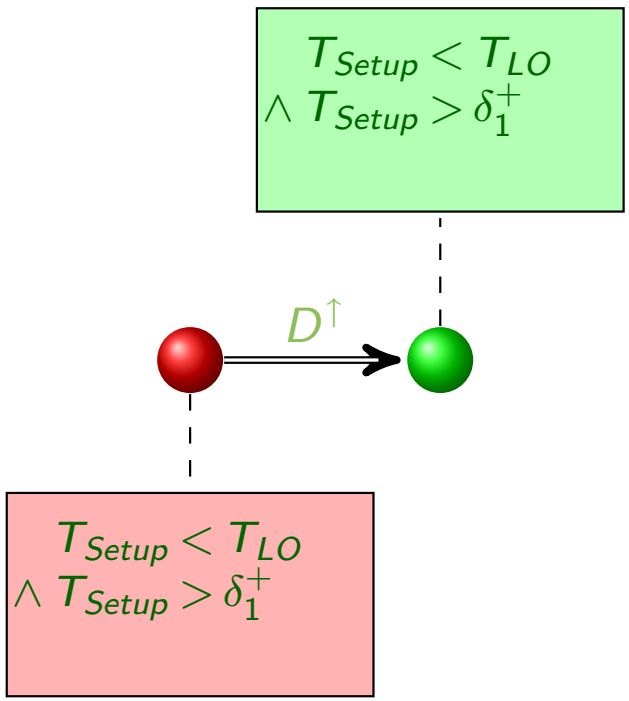
$T_{Setup} < T_{LO}$
$\wedge\ T_{Setup} > \delta_1^+$

Application à un exemple de circuit

π_0 :

$\delta_1^- = 1$	$\delta_1^+ = 1$	$T_{HI} = 20$
$\delta_2^- = 8$	$\delta_2^+ = 10$	$T_{LO} = 15$
$\delta_3^- = 5$	$\delta_3^+ = 6$	$T_{Setup} = 10$
$\delta_4^- = 3$	$\delta_4^+ = 5$	$T_{Hold} = 15$

$K_0 =$
$T_{Setup} < T_{LO}$
$\wedge\ T_{Setup} > \delta_1^+$

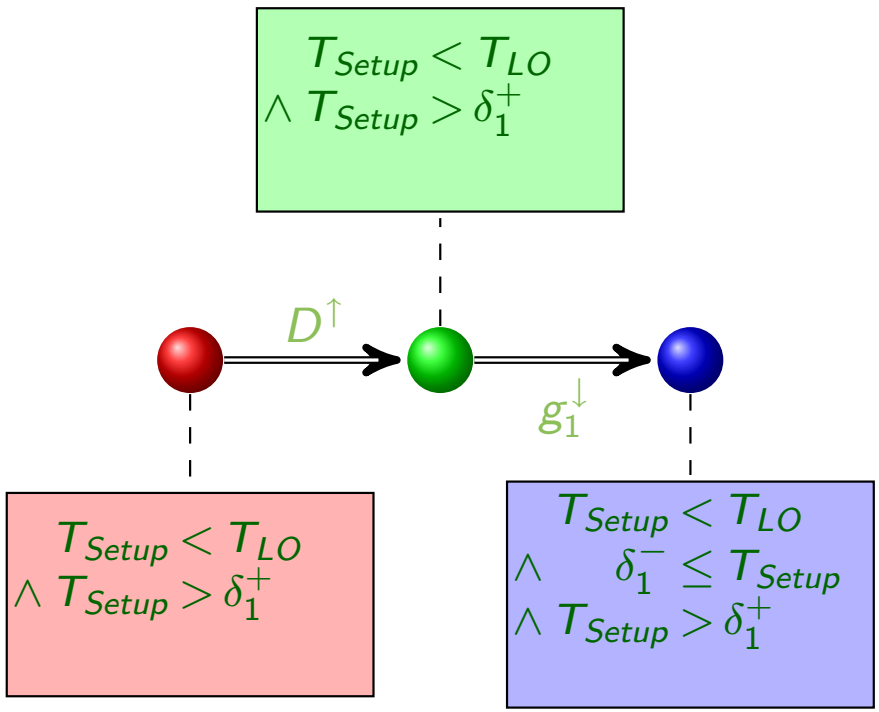


Application à un exemple de circuit

π_0 :

$\delta_1^- = 1$	$\delta_1^+ = 1$	$T_{HI} = 20$
$\delta_2^- = 8$	$\delta_2^+ = 10$	$T_{LO} = 15$
$\delta_3^- = 5$	$\delta_3^+ = 6$	$T_{Setup} = 10$
$\delta_4^- = 3$	$\delta_4^+ = 5$	$T_{Hold} = 15$

$K_0 =$
$T_{Setup} < T_{LO}$
$\wedge\ T_{Setup} > \delta_1^+$

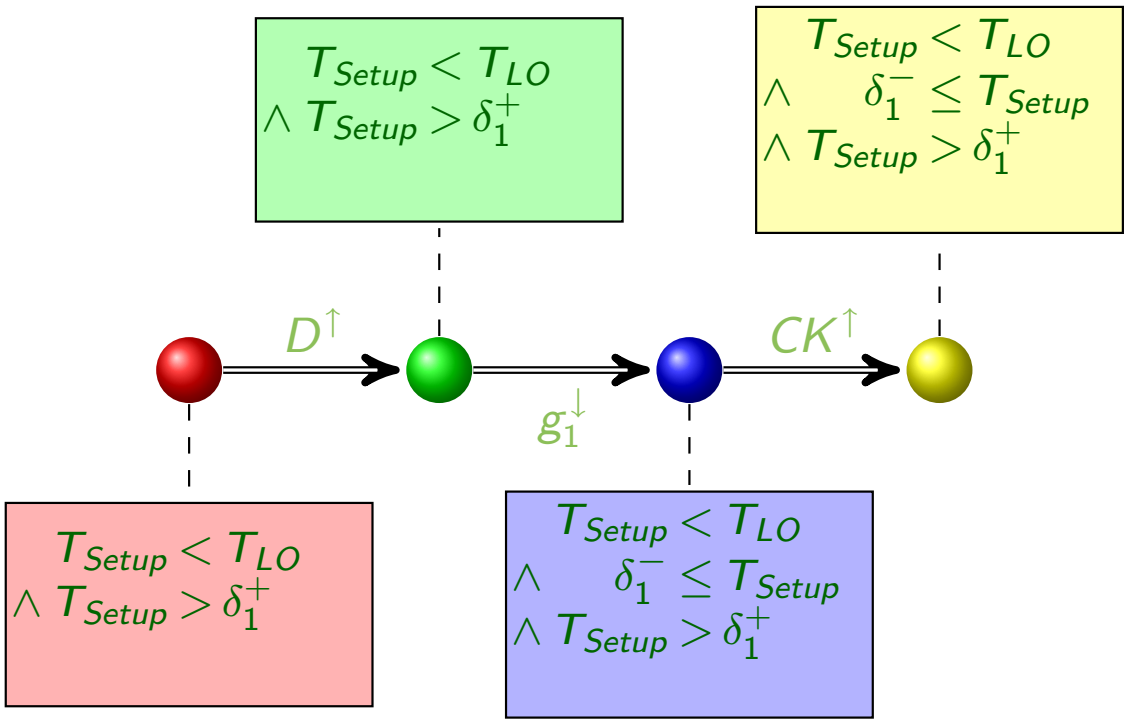


Application à un exemple de circuit

π_0 :

$\delta_1^- = 1$	$\delta_1^+ = 1$	$T_{HI} = 20$
$\delta_2^- = 8$	$\delta_2^+ = 10$	$T_{LO} = 15$
$\delta_3^- = 5$	$\delta_3^+ = 6$	$T_{Setup} = 10$
$\delta_4^- = 3$	$\delta_4^+ = 5$	$T_{Hold} = 15$

$K_0 =$
$T_{Setup} < T_{LO}$
$\wedge\ T_{Setup} > \delta_1^+$



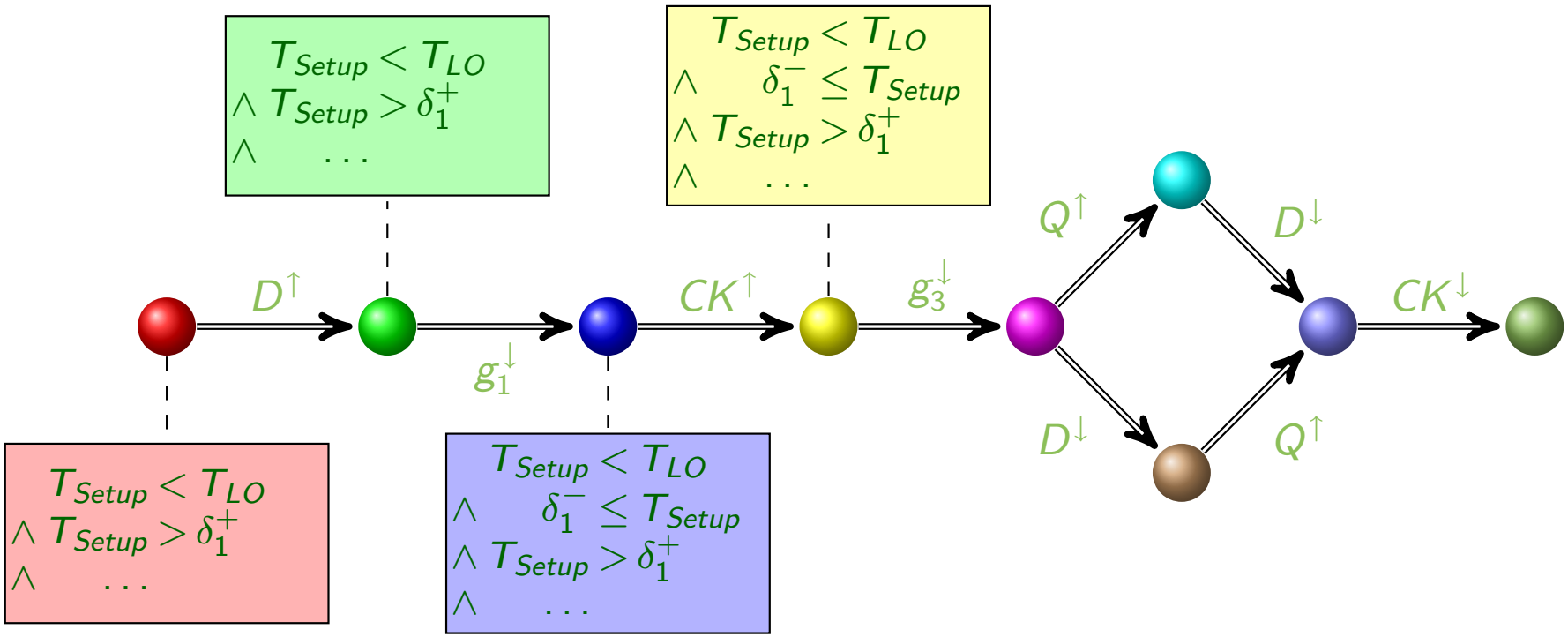
Application à un exemple de circuit

π_0 :

$\delta_1^- = 1$	$\delta_1^+ = 1$	$T_{HI} = 20$
$\delta_2^- = 8$	$\delta_2^+ = 10$	$T_{LO} = 15$
$\delta_3^- = 5$	$\delta_3^+ = 6$	$T_{Setup} = 10$
$\delta_4^- = 3$	$\delta_4^+ = 5$	$T_{Hold} = 15$

$K_0 =$

$$
\begin{array}{lll}
T_{Setup} < T_{LO} & \wedge & \delta_3^+ + \delta_4^+ < T_{HI} \\
\wedge\ T_{Setup} > \delta_1^+ & \wedge & \delta_3^+ + \delta_4^+ \geq T_{Hold} \\
\wedge\ \delta_3^+ < T_{Hold} & \wedge & \delta_3^- + \delta_4^- \leq T_{Hold} \\
\wedge\ \delta_1^- > 0 & &
\end{array}
$$



Plan

Implémentation

- IMITATOR [André, 2009]
  - IMITATOR : "Inverse Method for Inferring Time AbstracT BehaviOR"
  - 1500 lignes de code
  - 4 hommes-mois d'implémentation
  - Programme écrit en Python
  - Appels au model-checker paramétrique HYTECH

- IMITATOR est disponible sur sa page Web
  - `http://www.lsv.ens-cachan.fr/~andre/IMITATOR`

La contrainte synthétisée

- Application de l'outil IMITATOR sur notre système modélisé par des automates temporisés paramétrés
  - 5 automates, 8 horloges, 7 paramètres
  - 51 itérations, 956 états accessibles

La contrainte synthétisée

- Application de l'outil IMITATOR sur notre système modélisé par des automates temporisés paramétrés
  - 5 automates, 8 horloges, 7 paramètres
  - 51 itérations, 956 états accessibles

- Synthèse d'une contrainte K_0 sur les 7 paramètres du système

La contrainte synthétisée

- Application de l'outil IMITATOR sur notre système modélisé par des automates temporisés paramétrés
  - 5 automates, 8 horloges, 7 paramètres
  - 51 itérations, 956 états accessibles

- Synthèse d'une contrainte K_0 sur les 7 paramètres du système

- Projection de K_0 en 3 dimensions ($COMct$, $PLCct$ et $SIGmrt$) :
  - Instanciation de $PLCmtt$, $RIOd$, $COMd$ et $NETd$ par leur valeur telle que définie dans π_0
  - Obtention de la contrainte suivante :

$$
\begin{array}{rrcl@{\quad}rrcl}
 & SIGmrt & > & 2COMct + 70 & \wedge & COMct & > & 3PLCct + 90 \\
\wedge & 10PLCct & < & 3COMct + 10 & \wedge & 7PLCct & > & 2COMct + 90 \\
\wedge & 2COMct & < & 6PLCct + 205 & \wedge & 3COMct & < & 10PLCct + 10 \\
\wedge & 7PLCct & < & 2COMct + 105 & & & & \\
\end{array}
$$

Comparaison avec une méthode dichotomique

- Méthode 1 : exploration dichotomique [Ruel, 2009]
  - Principe : test d'un grand nombre de points grâce à un script appelant le model-checker UPPAAL
  - Avantage : vaste nuage de points de bon fonctionnement
  - Inconvénients : pas d'information sur les valeurs entre les points testés, seules 3 dimensions considérées (*COMct*, *PLCct* et *SIGmrt*)

- Méthode 2 : notre méthode de synthèse de contrainte
  - Avantage : zone dense en 7 dimensions
  - Inconvénient : zone plus petite que le nuage de points de la méthode 1

Plan

Récapitulatif

- Architecture d'automatisation en réseau
  - Modélisation par des automates temporisés paramétrés
  - 7 paramètres temporels
- Synthèse d'une contrainte K_0
  - À l'aide d'une valuation de référence π_0 des paramètres, application de la méthode inverse
  - Obtention d'une zone dense en 7 dimensions

Directions de recherches futures

- Amélioration de la contrainte K_0
  - La contrainte synthétisée par notre méthode n'est pas minimale
    - ⋆ La contrainte minimale peut n'exister que sous forme disjonctive
    - ⋆ La contrainte synthétisée par la méthode inverse garantit l'égalité des traces : plus restrictif que la non-accessibilité d'un « mauvais état »
    - ⋆ La contrainte synthétisée par la méthode inverse n'est elle même pas minimale
  - Objectif : minimisation de la contrainte K_0
    - ⋆ Méthode itérative tirant profit du nuage de points [André et al., 2009]
    - ⋆ Pour chaque point généré par la méthode dichotomique, rappeler notre méthode sur ce point, jusqu'à pavage de la zone entière
    - ⋆ Obtention d'une disjonction de contraintes couvrant une zone plus large
- Scalabilité
  - Étudier des systèmes plus grands (nombre de modules, nombre de paramètres)

Bibliographie


Alur, R. and Dill, D. L. (1994).
A theory of timed automata.
TCS, 126(2) :183–235.


Alur, R., Henzinger, T. A., and Vardi, M. Y. (1993).
Parametric real-time reasoning.
In *STOC '93*, pages 592–601, New York, USA. ACM.


André, É. (2009).
IMITATOR : A tool for synthesizing constraints on timing bounds of timed automata.
In Leucker, M. and Morgan, C., editors, *ICTAC'09*, volume 5684 of *Lecture Notes in Computer Science*, pages 336–342, Kuala Lumpur, Malaysia. Springer.


André, É., Chatain, T., Encrenaz, E., and Fribourg, L. (2009).
An inverse method for parametric timed automata.
International Journal of Foundations of Computer Science, 20(5) :819–836.


Ruel, S. (2009).
Évaluation des bornes des performances temporelles des Architectures d'Automatisation en Réseau par preuves itératives de propriétés logiques.
Thèse de doctorat, ENS Cachan, France.