

Des données aux modèles hybrides

Olivier Roux

Ecole Centrale de Nantes / LS2N

B.P. 92101, 44321 NANTES cedex 3, (FRANCE)

[tel: (33)0 2.40.37.69.79 -- e-mail: olivier.roux@ec-nantes.fr]

Le travail présenté ici a profité des contributions de G. Bernot, J.P. Comet, M. Folschette, M. Magnin, L. Paulevé, T. Ribeiro, ...

— MODÉLISATION FORMELLE DE RÉSEAUX DE RÉGULATION BIOLOGIQUE — Ile de Porquerolles — juin 2019 —



Sommaire

- 1 Introduction
- 2 Généralités — sur la représentation du temps dans les modèles de la dynamique des système
- 3 »»» PREMIERE PARTIE : Des modèles hybrides «««
- 4 Model-checking temporisé
- 5 Model-checking hybride
- 6 Abstraction et raffinement
- 7 »»» DEUXIEME PARTIE : Des données aux modèles «««
- 8 Apprentissage des modèles
- 9 Extension hybride : du discret ... aux intervalles
- 10 Conclusion
- 11 Bibliographie

"Des données aux modèles"... quel genre de modèle ?

Une modélisation continue ?

Une modélisation complètement discrète ?

"Des données aux modèles"... quel genre de modèle ?

Une modélisation continue ?

Une modélisation complètement discrète ?

Voyons sur un exemple :

Pseudomonas *æ*ruginosa

Pathogène nosocomial majeur, en particulier chez les patients atteints de mucoviscidose et dans les services de réanimation. L'augmentation actuelle de l'incidence des souches multirésistantes de *P. æ*ruginosa (PAMR) et les phénomènes épidémiques locaux qui en résultent sont donc particulièrement inquiétants.

"Des données aux modèles"... quel genre de modèle ?

Modélisation de la production de mucus chez PA [BCRG04, ABC⁺07]

In this section we present R. Thomas' modeling of genetic regulatory networks through the example of the mucus production system in *Pseudomonas aeruginosa*, an opportunistic pathogen, which is often encountered in chronic lung diseases such as cystic fibrosis. These bacteria are commonly present in the environment and secrete mucus only in lungs affected by cystic fibrosis. As this mucus increases the respiratory deficiency of the patient, it is the major cause of mortality. The regulatory network which controls the mucus production has been elucidated [GMK01]. The main regulator for the mucus production, AlgU, supervises an operon which is made of 4 genes among which one codes for a protein that is an inhibitor of AlgU. Moreover AlgU favors its own synthesis. The mucus production regulatory network can then be simplified into the regulatory graph of Figure 2, where **node 1** represents AlgU, and **node 2** its inhibitor.

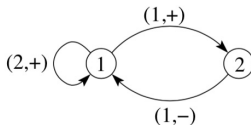


Figure 2 : Regulatory graph for mucus production in *Pseudomonas aeruginosa*

"Des données aux modèles"... quel genre de modèle ?

Modélisation de la production de mucus chez PA [BCRG04, ABC+07]

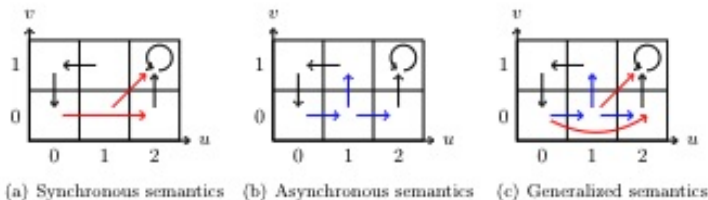


Figure 2.3: State transition graphs of different updating schemes

"Des données aux modèles"... quel genre de modèle ?

Modélisation de la production de mucus chez PA [BCRG04, ABC⁺07]

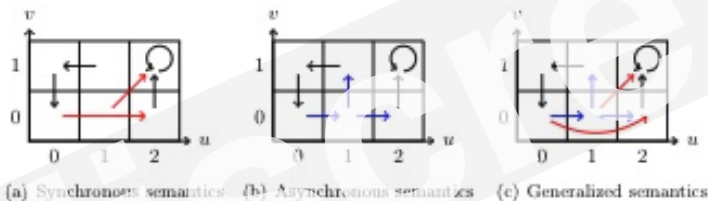


Figure 2.3: State transition graphs of different updating schemes

"Des données aux modèles"... quel genre de modèle ?

Modélisation de la production de mucus chez PA [BCRG04, ABC⁺07]

The interaction graph abstracting the mucus production system in *Pseudomonas aeruginosa* contains a positive feedback circuit. This makes possible a dynamic with two stationary states which would allow, from a biological point of view, an epigenetic change (stable change of phenotype without mutation) from the non mucoid state to the mucoid one. The dynamical hypothesis (epigenetic hypothesis) can be translated into a CTL specification [FMB⁺06]: stability of mucoid state (resp. non mucoid state) is expressed by $(x = 2) \Rightarrow AXAF(x = 2)$ (resp. $(x = 0) \Rightarrow AG(\neg(x = 2))$). These formulae mean that, if x is equal to 2, then it will ever be equal to 2 in the future, and (resp.) that, if x is equal to 0, then it will never be equal to 2. Intensive model checking approach, implemented in SMBioNet¹ [BCRG04], formally proves that the epigenetic hypothesis is consistent because 8 models satisfy these formulae from which one is presented in Figure 4-(a).

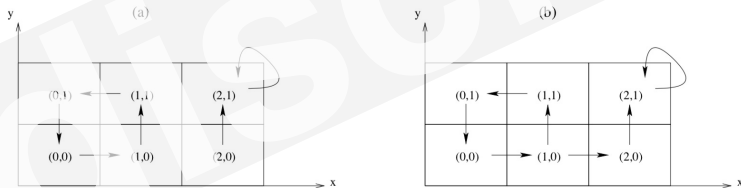


Figure 4 : Two asynchronous state graphs for mucus production in *Pseudomonas aeruginosa*.

(a) validates the CTL specification whereas (b) does not

"Des données aux modèles"... quel genre de modèle ?

Modélisation de la production de mucus chez PA [BCRG04, ABC⁺07]

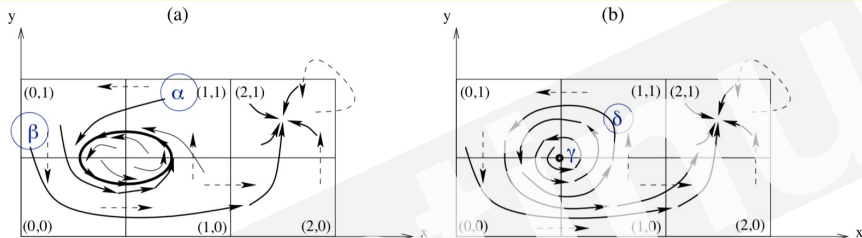


Figure 5 : Continuous trajectories compatible with asynchronous state graphs of Figure 4-b.

Cyclic trajectory (a), and divergent trajectory (b)

We introduce in this paper delays in the modeling to separate these different classes of behaviours. It is expected that *under certain constraints on delay parameters*, the model presents two **cyclic trajectory**, representing respectively the mucoid **state**, and **one attractive basin representing** the non mucoid state. Under other constraints, the system presents only one **attractive basin**. Biological knowledge is then taken into consideration to choose between the models with

"Des données aux modèles"... quel genre de modèle ?

Modélisation de la production de mucus chez PA [BCRG04, ABC⁺07]

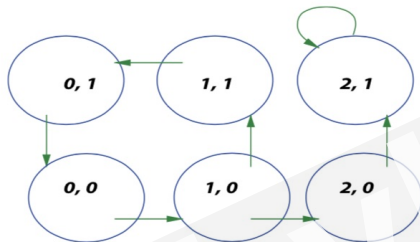


Fig. 7. Example of a GRN discrete model (see fig. 4b).

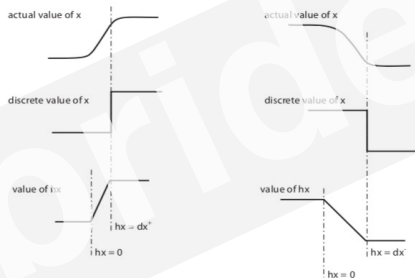
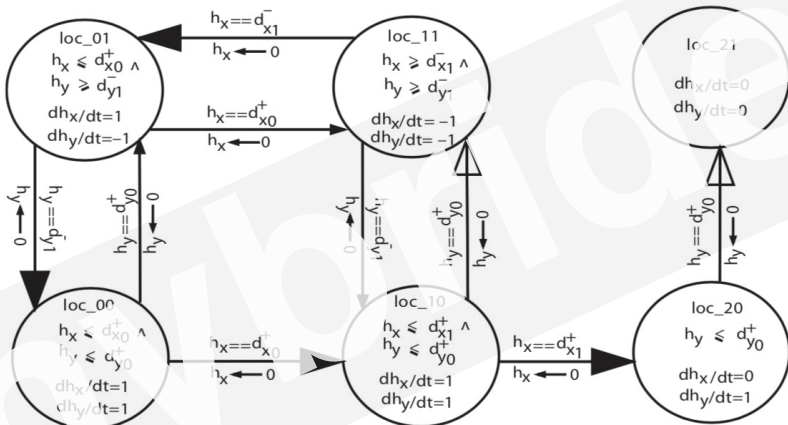


Fig. 8. Modelling of time delays.

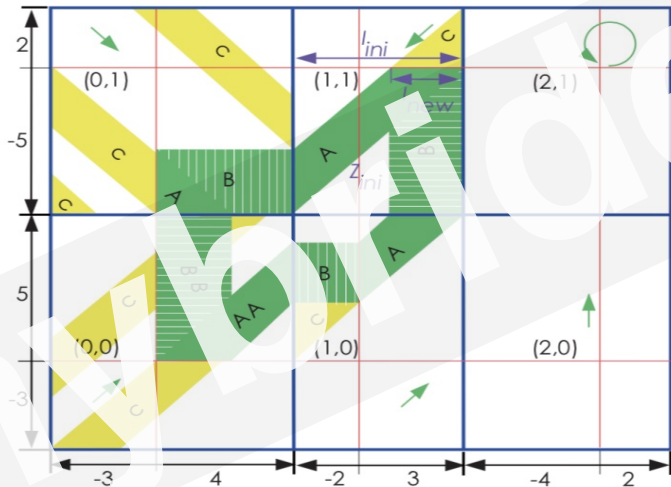
"Des données aux modèles"... quel genre de modèle ?

Modélisation de la production de mucus chez PA [BCRG04, ABC⁺07]



"Des données aux modèles"... quel genre de modèle ?

Modélisation de la production de mucus chez PA [BCRG04, ABC⁺07]



Objectifs de ce cours

- Pour étudier la *dynamique* d'un système

Objectifs de ce cours

- Pour étudier la *dynamique* d'un système
- Comment modéliser ?

Objectifs de ce cours

- Pour étudier la *dynamique* d'un système
- Comment modéliser ?
- Introduire le temps dans le modèles : *chronologie*, *chronométrie*
 ↔ *une modélisation hybride* **MAIS** ...

Objectifs de ce cours

- Pour étudier la *dynamique* d'un système
- Comment modéliser ?
- Introduire le temps dans le modèles : *chronologie*, *chronométrie*
 \rightsquigarrow *une modélisation hybride* **MAIS** ...
- Affiner et abstraire : la convergence des approximations

Objectifs de ce cours

- Pour étudier la *dynamique* d'un système
- Comment modéliser ?
- Introduire le temps dans le modèles : *chronologie*, *chronométrie*
 \rightsquigarrow *une modélisation hybride* **MAIS** ...
- Affiner et abstraire : la convergence des approximations
 \rightsquigarrow *Des données aux modèles...*

Objectifs de ce cours (suite...)

- transmettre quelques techniques

plus ou moins classiques...

Objectifs de ce cours (suite...)

- transmettre quelques techniques

plus ou moins classiques...

- faire passer quelques idées

plus ou moins nouvelles...

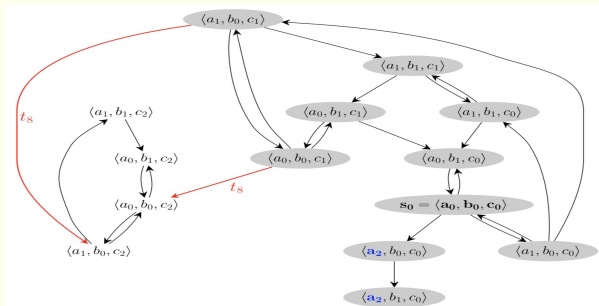
Sommaire

- 1 Introduction
- 2 **Généralités** — sur la représentation du temps dans les modèles de la dynamique des système
- 3 »»» PREMIERE PARTIE : Des modèles hybrides «««
- 4 Model-checking temporisé
- 5 Model-checking hybride
- 6 Abstraction et raffinement
- 7 »»» DEUXIEME PARTIE : Des données aux modèles «««
- 8 Apprentissage des modèles
- 9 Extension hybride : du discret ... aux intervalles
- 10 Conclusion
- 11 Bibliographie

Généralités — sur la représentation du temps dans les modèles de la dynamique des système

Des propriétés intéressantes sur le fonctionnement du système :

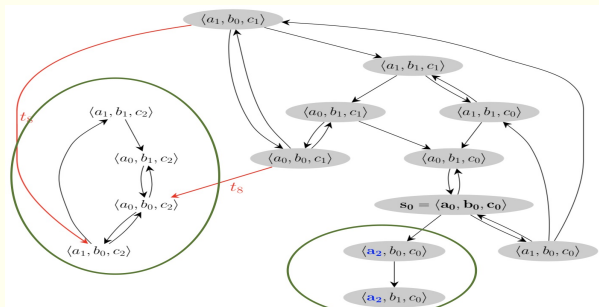
- Accessibilité
- Points fixes
- Attracteurs, bassins d'attraction, ...
- Bifurcations
- Cut-sets
- ...



Généralités — sur la représentation du temps dans les modèles de la dynamique des système

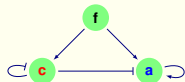
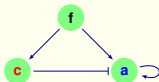
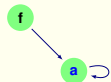
Des propriétés intéressantes sur le fonctionnement du système :

- Accessibilité
- Points fixes
- Attracteurs, bassins d'attraction, ...
- Bifurcations
- Cut-sets
- ...



Introduire le temps :

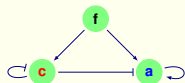
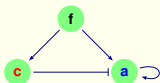
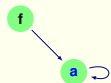
Un autre exemple (Metazoan segmentation [P. Francois])



▶ retour

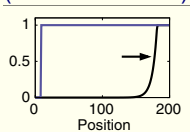
Introduire le temps :

Un autre exemple (Metazoan segmentation [P. Francois])



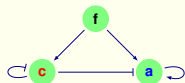
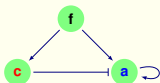
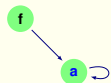
[▶ retour](#)

(activation.mov)



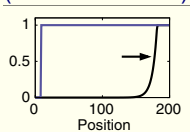
Introduire le temps :

Un autre exemple (Metazoan segmentation [P. Francois])

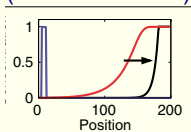


▶ retour

(activation.mov)

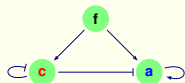
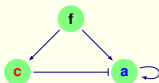
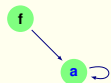


(activation.mov)



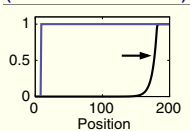
Introduire le temps :

Un autre exemple (Metazoan segmentation [P. Francois])

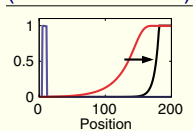


► retour

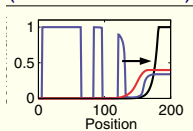
(activation.mov)



(activation.mov)



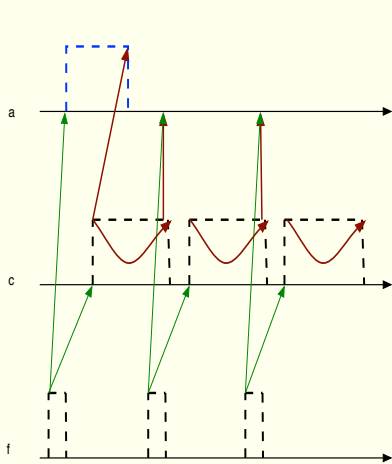
(activation.mov)



Introduire le temps (suite)

Différents comportements (suivant les durées) :

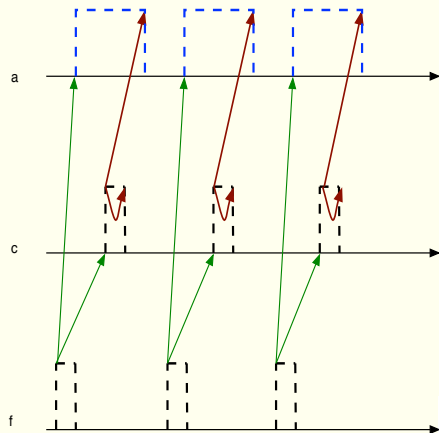
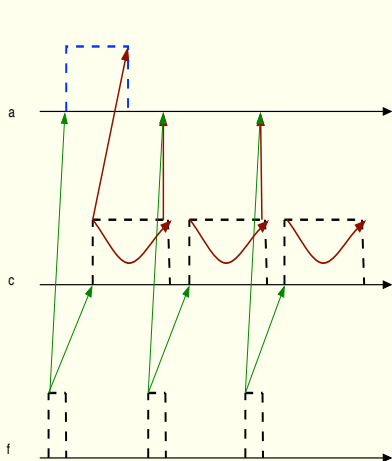
$$a \Leftarrow f \wedge \neg c \quad \text{et} \quad c \Leftarrow f$$



Introduire le temps (suite)

Différents comportements (suivant les durées) :

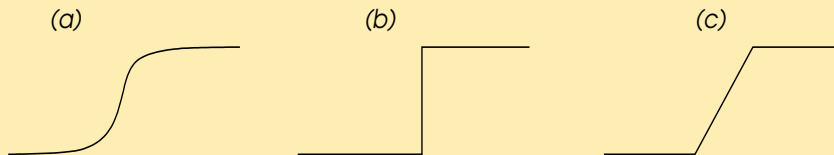
$$a \Leftarrow f \wedge \neg c \quad \text{et} \quad c \Leftarrow f$$



(H) Modélisation hybride

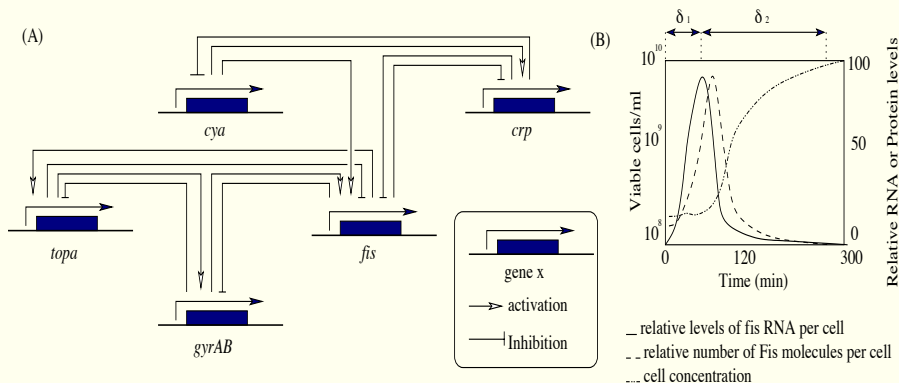
Prise en compte du temps : les délais pour les changements de niveau qui ne sont plus considérés comme instantanés

Figure 1 : Une sigmoïde (a), sa caricature logique (b) et sa caricature linéaire par morceaux (c)



Les valeurs relatives de délais et retards permettent de discriminer les comportements

Figure 2 : Exemple de la bactérie Escherichia coli



[Ropers et al.]

Pour introduire le temps :

Deux approches... + une

pour la dualité continu / discret

▷ Automates temporisés :

Une approche états-transitions introduisant des "*horloges*"

↪ model-checking **temporisés**.

▷ Automates hybrides :

Une approche états-transitions prenant en compte les évolutions temporelles de variables progressant avec différentes vitesses

↪ model-checking **hybride**.

Pour introduire le temps :

Deux approches... + une

pour la dualité continu / discret

▷ Automates temporisés :

Une approche états-transitions introduisant des "*horloges*"

↪ model-checking **temporisés**.

▷ Automates hybrides :

Une approche états-transitions prenant en compte les évolutions temporelles de variables progressant avec différentes vitesses

↪ model-checking **hybride**.

▷ L'approche des célérités ...

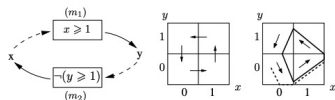


Fig. 1. (Left) Graphical representation of a gene regulatory graph with multiplexes. Dashed lines represent participation of variables in multiplexes (not present in Definition 1). (Center) State graph obtained with parameters $K_{x,\{1\}} = 0$, $K_{x,\{m_2\}} = 1$, $K_{y,\{1\}} = 0$, $K_{y,\{m_1\}} = 1$. (Right) Hybrid trajectories obtained by the following celerities: $C_{x,\{1\},0} = -1.5$, $C_{x,\{1\},1} = -4$, $C_{x,\{m_2\},0} = 1.5$, $C_{x,\{m_2\},1} = 4$, $C_{y,\{1\},0} = -3.33$, $C_{y,\{1\},1} = -3$, $C_{y,\{m_1\},0} = 2.4$ and $C_{y,\{m_1\},1} = 3$.

...
 et je ne parlerai pas non plus d'une autre approche : les Réseaux de Petri hybrides...

e.g.

UNIVERSITÉ D'ÉVRY-VAL D'ESSONNE
 U.F.R. SCIENCES FONDAMENTALES ET APPLIQUÉES

THÈSE

présentée pour obtenir

LE GRADE DE DOCTEUR EN SCIENCES
 DE L'UNIVERSITÉ D'ÉVRY-VAL D'ESSONNE

Spécialité
 BIOINFORMATIQUE

Sylvie TRONCALE

MODÉLISATION ET VÉRIFICATION DES RÉSEaux DE PETRI
 HYBRIDES TEMPORISÉS.

- APPLICATION À LA MÉTAMORPHOSE AMPHIBIENNE -

Soutenu le 12 septembre 2008 devant le jury composé de

M. Gilles BERNOT	<i>Directeur de thèse</i>
M. Jean-Paul COMET	<i>Directeur de thèse</i>
M. Alexander BOCKMAYR	<i>Rapporteur</i>
M. François FAGES	<i>Rapporteur</i>
M. Bruno BOST	<i>Examinateur</i>
M. Nicolas POLLET	<i>Examinateur</i>
M. Philippe TRACQUI	<i>Examinateur</i>

Thèse préparée au sein du laboratoire IBISC
 et du Programme d'Épigénomique
 CNRS / Génopole / Université d'Évry-Val d'Essonne

Sommaire

- 1 Introduction
- 2 Généralités — sur la représentation du temps dans les modèles de la dynamique des système
- 3 »»» PREMIERE PARTIE : Des modèles hybrides «««
- 4 Model-checking temporisé
- 5 Model-checking hybride
- 6 Abstraction et raffinement
- 7 »»» DEUXIEME PARTIE : Des données aux modèles «««
- 8 Apprentissage des modèles
- 9 Extension hybride : du discret ... aux intervalles
- 10 Conclusion
- 11 Bibliographie

Sommaire

- 1 Introduction
- 2 Généralités — sur la représentation du temps dans les modèles de la dynamique des système
- 3 »»» PREMIERE PARTIE : Des modèles hybrides «««
- 4 Model-checking temporisé**
- 5 Model-checking hybride
- 6 Abstraction et raffinement
- 7 »»» DEUXIEME PARTIE : Des données aux modèles «««
- 8 Apprentissage des modèles
- 9 Extension hybride : du discret ... aux intervalles
- 10 Conclusion
- 11 Bibliographie

Related works

■ Automates temporisés

- ▶ [\[M. Adélaïde and G. Sutre\]](#). "Parametric Analysis and Abstraction of Genetic Regulatory Networks". In *Concurrent Models in Molecular Biology (BioCONCUR'04)*, London, UK, Aug. 2004, *Electronic Notes in Theor. Comp. Sci., Elsevier, 2004*
- ▶ [\[H. Sieber and A. Bockmayr\]*](#). "Incorporating Time Delays into the Logical Analysis of Gene Regulatory Networks". In *Computational Methods in Systems Biology, (CMSB 2006)*
- ▶ [\[O. Maler, G. Batt\]](#). "Approximating Continuous Systems by Timed Automata", In *Formal Methods in Systems Biology, 2008*
- ▶ [\[A. David, K. Larsen, A. Legay, M. Mikucionis, D. Poulsen, et al.\]](#). "Runtime Verification of Biological Systems". In *Leveraging Applications of Formal Methods, Verification and Validation. Technologies for Mastering Change*, 2012. LNCS 7609, pp.388-404
- ▶ [\[S. Schivo, J. Scholma, B. Wanders, R. Urquidi Camacho, P. Van der Pol\]](#) [\[Vet, M. Karperien, R. Langerak, J. Van de Pol, J. Post\]](#). "Modeling biological pathway dynamics with timed automata" In *IEEE journal of biomedical and health informatics 18(3) pp. 832-839, 2013*
- ▶ [\[R. Langerak, J. van de Pol, J. N. Post, and S. Schivo\]](#). "Improving the Timed Automata Approach to Biological Pathway Dynamics" In *Models, Algorithms, Logics and Tools. (LNCS, vol. 10460) pp. 96-111, 2017*

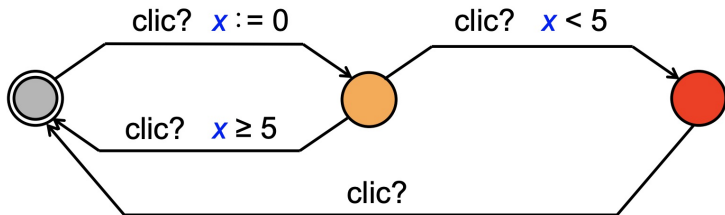


Commençons par un exemple [une lampe à double allumage]

[D'après les supports du cours de Gérard Berry au Collège de France
"mes transparents sont mis à disposition et ils sont faits pour être volés"]



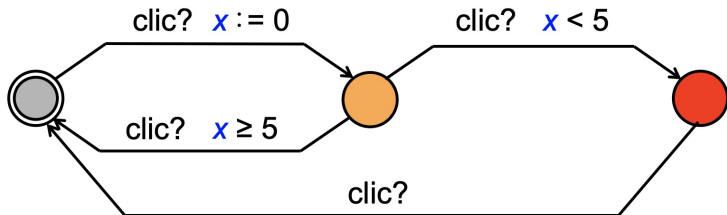
Automates temporisés : lampe à double-clic



[AD] Rajeev Alur et Davis Dill, *A Theory of Timed Automata*

Commençons par un exemple [une lampe à double allumage]

Automates temporels : lampe à double-click



0 :

0.4 : clic? → ● $x := 0$

7.3 : clic? $x == 6.9$ → ●

0 :

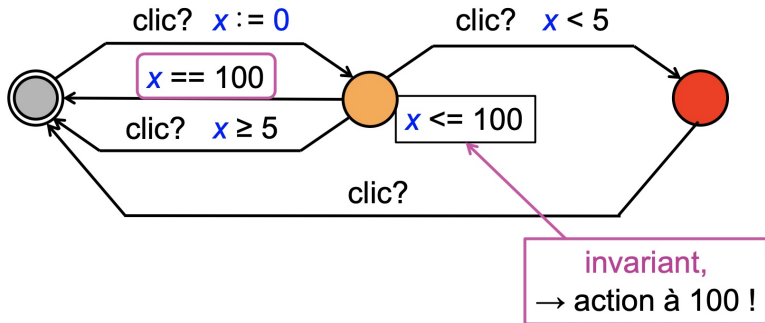
0.4 : clic? → ● $x := 0$

4.3 : clic? $x == 3.9$ → ●

7.6 : clic? $x == 7.2$ → ●

Commençons par un exemple [une lampe à double allumage]

La même lampe avec minuterie partielle



0 :

0.4 : clic? → `x := 0` ●

100.4 : `x == 100` → ●

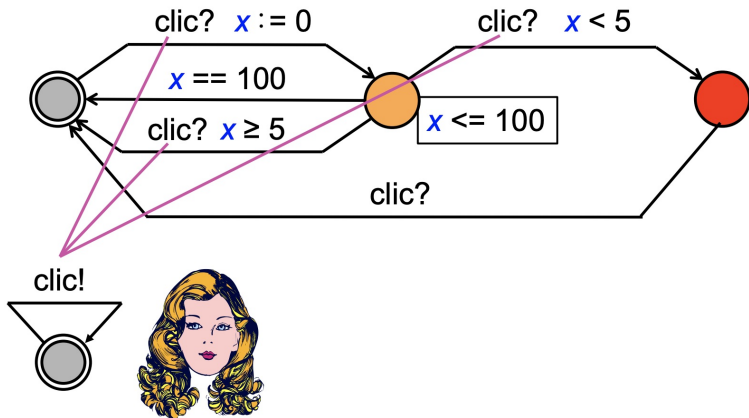
0 :

0.4 : clic? → ● `x := 0`

4.3 : clic? → ● `x == 3.9`

142.7 : clic? → ● `x == 142.3`

Commençons par un exemple [une lampe à double allumage]



Réseau d'automates à la CCS, descriptif et pas prescriptif :
action = rendez-vous clic!-clic? instantané mais asynchrone

Définition : Automate temporisé[▶ voir la définition des automates hybrides](#)

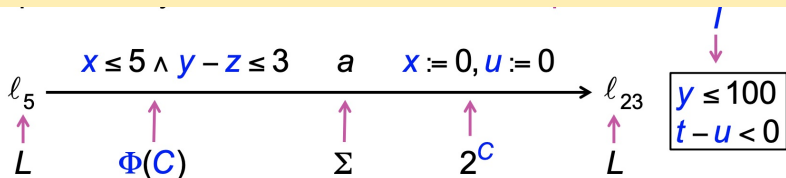
- Un automate temporisé A est défini par $A = (L, \ell_0, C, \Sigma, I, E)$ où :
 1. $L = \{\ell_i\}$ est un ensemble fini de *places*
 2. ℓ_0 est la *place initiale*
 3. C est l'ensemble des *horloges*
 4. Σ est l'ensemble des *actions*
 4. $I: L \rightarrow \Phi(C)$ est l'ensemble des *invariants de places*
 5. $E \subseteq L \times \Phi(C) \times \Sigma \times 2^C \times L$ est l'ensemble des *flèches*

Définition : Automate temporisé

→ voir la définition des automates hybrides

- Un automate temporisé A est défini par $A = (L, \ell_0, C, \Sigma, I, E)$ où :
 - $L = \{\ell_i\}$ est un ensemble fini de *places*
 - ℓ_0 est la *place initiale*
 - C est l'ensemble des *horloges*
 - Σ est l'ensemble des *actions*
 - $I: L \rightarrow \Phi(C)$ est l'ensemble des *invariants de places*
 - $E \subseteq L \times \Phi(C) \times \Sigma \times 2^C \times L$ est l'ensemble des *flèches*

Transitions dans un automate temporisé :



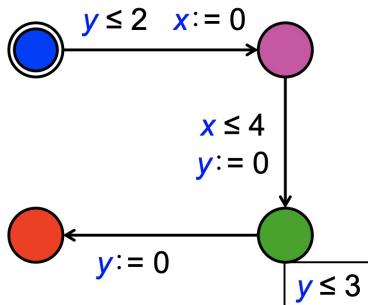
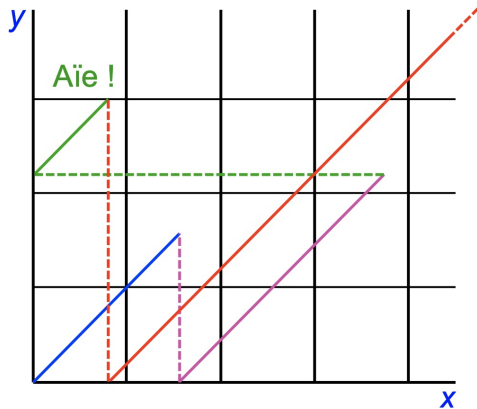
Transitions et comportement temporel

- Il y a deux sortes de **transitions** :
 1. **Passage du temps** sans bouger, en préservant les invariants :

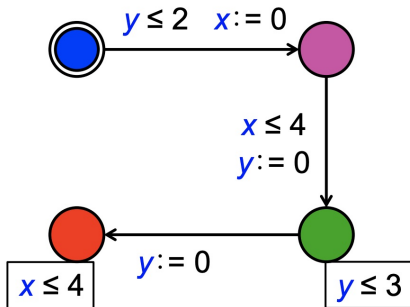
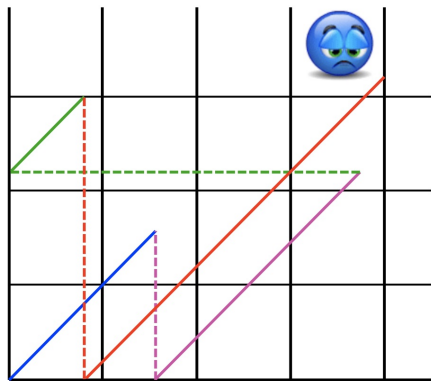
$$(l, v) \rightarrow (l, v + d)$$
 pour tout v et d tel que v et $v + d$ satisfont $I(l)$
 2. **Prise d'une flèche** autorisée

$$(l, v) \rightarrow (l', v')$$
 s'il y a une transition de la forme $l, I \xrightarrow{\phi, a, r} l', I'$
 telle que v satisfait I et ϕ et $v' = v[r := 0]$ satisfait I'
- Un **comportement temporel** est simplement une suite de transitions légales

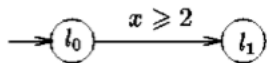
Exemples de comportements temporels

Comportements bi-horloges

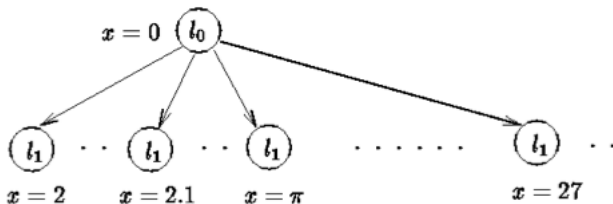
Exemples de comportements temporels

Comportements bi-horloges

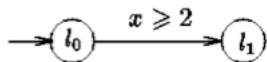
Les automates temporisés : Un espace d'états infini ?



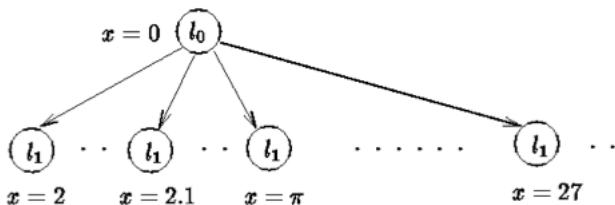
gives rise to the
infinite transition system:



Les automates temporisés : Un espace d'états infini ?



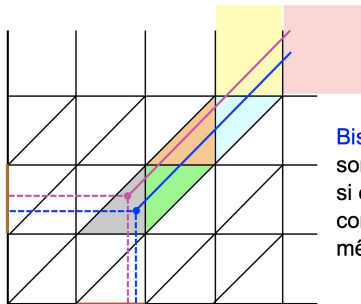
gives rise to the
infinite transition system:



Non,

car il existe des classes d'équivalence de beaucoup de ces états

Méthode pour la vérification (suite)

Bisimulation et régions

Régions :
point, bord de triangle
triangle, colonnes, etc.

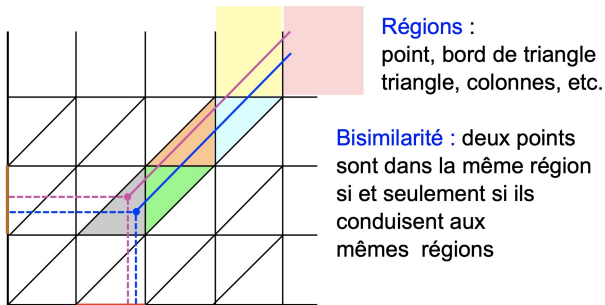
Bisimilarité : deux points
sont dans la même région
si et seulement si ils
conduisent aux
mêmes régions

Décidabilité et complexité

Pour un automate donné, on n'a qu'un nombre fini de régions

a. où c_x est la valeur maximale de contrainte de l'horloge x

Méthode pour la vérification (suite)

Bisimulation et régions

Décidabilité et complexité

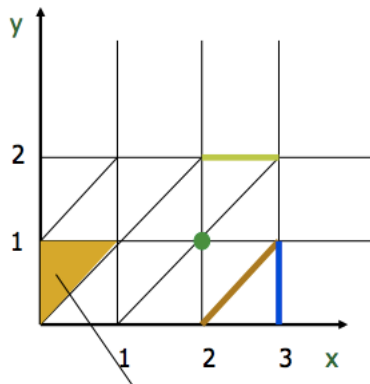
Pour un automate donné, on n'a qu'un nombre fini de régions

Mais le nombre de régions est exponentiel dans le nombre d'horloges :
[Alur & Dill] \rightsquigarrow pour n horloges x , borne sup = $n! \times 2^n \times \prod_x (2 \cdot c_x + 2)^a$

a. où c_x est la valeur maximale de contrainte de l'horloge x

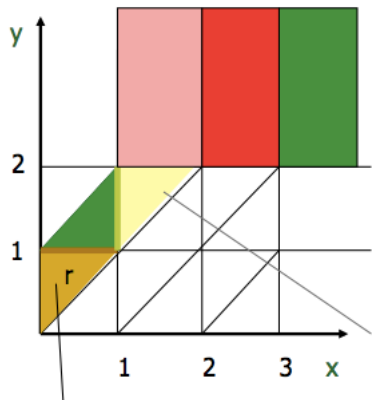
Les automates temporisés : partitionnement fini de l'espace des états

en régions (classes d'équivalence)



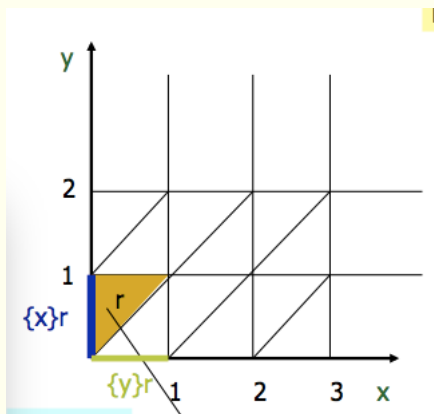
Définition : $\mathcal{R} \equiv \mathcal{R}'$ ssi \mathcal{R} et \mathcal{R}' satisfont exactement les mêmes conditions de la forme $x_i \leq n$ et $x_i - x_j \leq n$ où $n \leq \max$.

Les automates temporisés : partitionnement fini de l'espace des états en régions (classes d'équivalence) (suite)



Définition : $\mathcal{R} \equiv \mathcal{R}'$ ssi \mathcal{R} et \mathcal{R}' satisfont exactement les mêmes conditions de la forme $x_i \leq n$ et $x_i - x_j \leq n$ où $n \leq \max$.

Les automates temporisés : partitionnement fini de l'espace des états en régions (classes d'équivalence) (suite)



Définition : $\mathcal{R} \equiv \mathcal{R}'$ ssi \mathcal{R} et \mathcal{R}' satisfont exactement les mêmes conditions de la forme $x_i \leq n$ et $x_i - x_j \leq n$ où $n \leq \max$.

La logique temporelle de UPPAAL = $CTL^+ \equiv TCTL^-$

- Quantification d'états : **A**= tous, **E**=existe
- Quantification de chemins : **G**=toujours, **F**=un jour, **U**=jusqu'à
- Logique CTL (*Computation Tree Logic*)
expressions d'états, **AG** ϕ , **AF** ϕ , **EG** ϕ , **EF** ϕ , **E**(ϕ **U** ϕ')
- Logique d'UPPAAL : CTL restreinte + temps \rightarrow **décidable**
 - E** $\langle \rangle$ ϕ : **EF** ϕ , possible = il existe un chemin où ϕ sera vrai
 - A**[] ϕ : **AG** ϕ , toujours, abréviation de \neg **E** $\langle \rangle$ $\neg\phi$
 - E**[] ϕ : **EG** ϕ , possiblement toujours
 - A** $\langle \rangle$ ϕ : **AF** ϕ , toujours un jour, abréviation de \neg **E**[] $\neg\phi$
 - $\phi \rightsquigarrow \phi'$: **AG** ($\phi \Rightarrow$ **AF** ϕ')
si ϕ devient vrai, ϕ' deviendra toujours (vrai un jour)

Les horloges peuvent apparaître dans ϕ

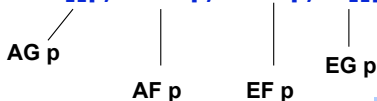
Introduction au logiciel UPPAAL — CTL et TCTL

TCTL Quantifiers in UPPAAL

- E - exists a path (“**E**” in UPPAAL).
- A - for all paths (“**A**” in UPPAAL).
- G - all states in a path (“**[**” in UPPAAL).
- F - some state in a path (“**<>**” in UPPAAL).

You may write the following queries in UPPAAL:

- **A[]p, A<>p, E<>p, E[]p and p --> q**

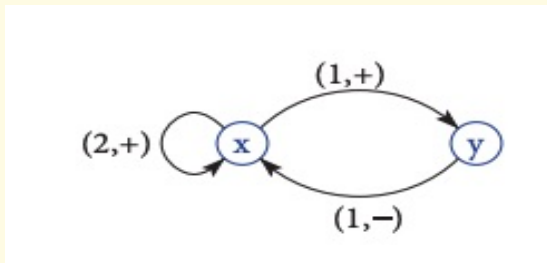


p and q are "local properties"

36

Model-checking temporisé

Retour sur la modélisation de *Pseudomonas aeruginosa*...
avec les automates temporisés

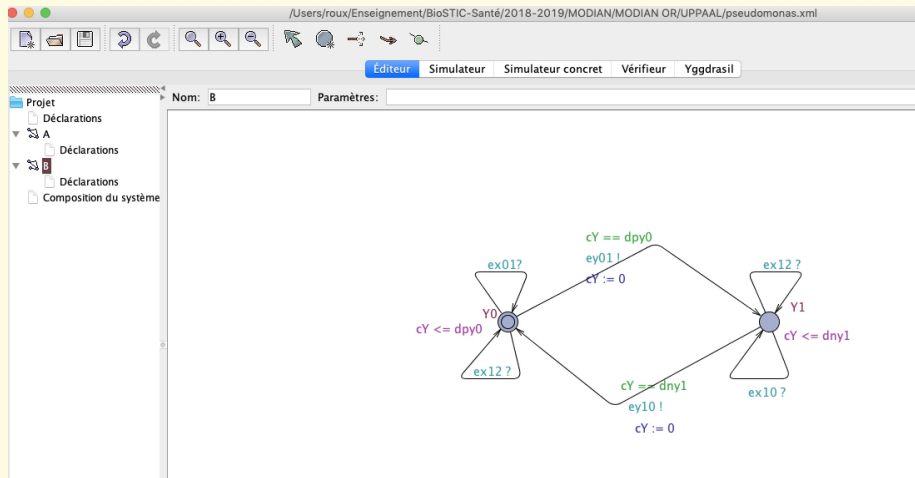


- ▷ quand y est au niveau 1, sous l'influence de $x \geq 1$, il va décroître en moins de $dn1$ unités de temps,
- ▷ ...

Model-checking temporisé

La représentation sous UPPAAL des comportements chez *Pseudomonas aeruginosa*

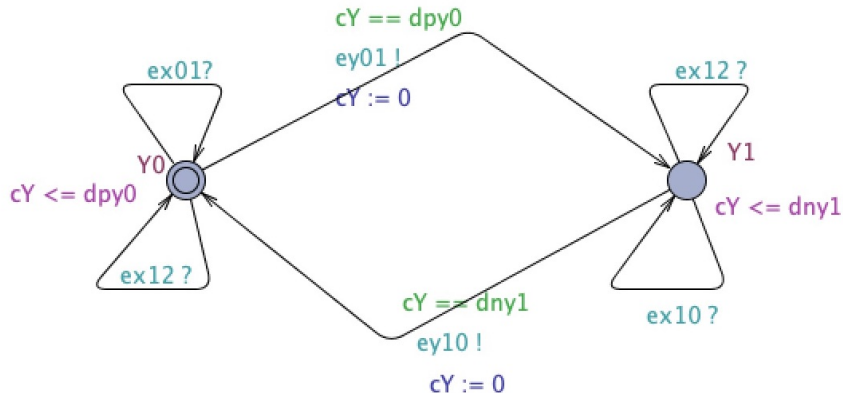
Modélisation du gène γ (2 états (0 et 1))



Model-checking temporisé

La représentation sous UPPAAL des comportements chez *Pseudomonas aeruginosa*

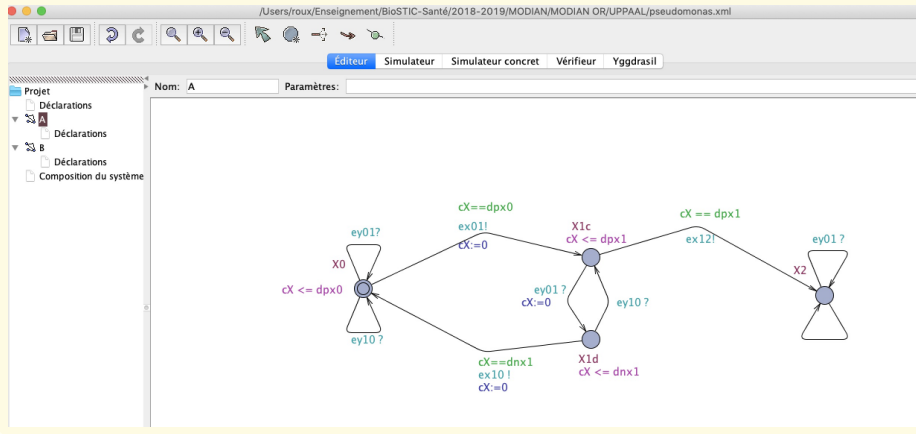
Modélisation du gène y (2 états (0 et 1))



Model-checking temporisé

La représentation sous UPPAAL des comportements chez *Pseudomonas aeruginosa*

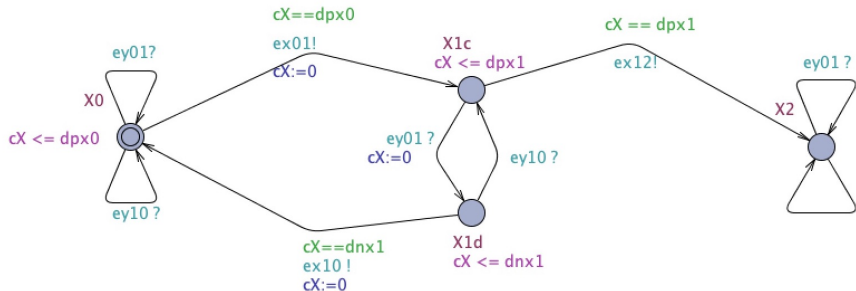
Modélisation du gène x (3 états)



Model-checking temporel

La représentation sous UPPAAL des comportements chez *Pseudomonas aeruginosa*

Modélisation du gène x (3 états)



Model-checking temporisé

La représentation sous UPPAAL des comportements chez *Pseudomonas aeruginosa*

Simulation du système synchronisant les deux gènes

Éditeur **Simulateur** Simulateur concret Vérifieur Yggdrasil

Transitions actives

```
ex10 : A → B
ey10 : B → A
```

▶ Suivant ↺ Reset

Trace de la simulation

```
(X1c, Y0)
ey01 : B → A
(X1d, Y1)
ex10 : A → B
(X0, Y1)
ey10 : B → A
(X0, Y0)
ex01 : A → B
```

Fichier de trace: _____

⏪ Préc. ⏸ Suv. ▶ Rejouer

📄 Ouvrir 📁 Enreg. ⏪ Auto

<Global variables>

```
dpx0 = 2
dnx1 = 2
dpx1 = 7
dpy0 = 3
dny1 = 2
```

<Constraints>

```
A.CX = 2
B.cY = 2
A.CX = B.cY
```

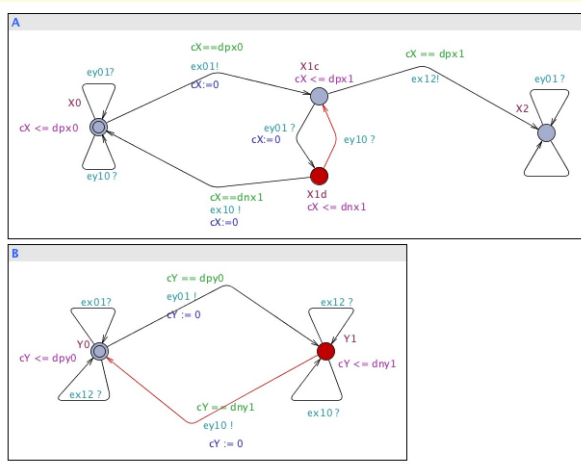
A

B

Model-checking temporisé

La représentation sous UPPAAL des comportements chez *Pseudomonas aeruginosa*

Simulation du système synchronisant les deux gènes



Model-checking temporisé

La représentation sous UPPAAL des comportements chez *Pseudomonas aeruginosa*

Vérification de la propriété d'accessibilité de l'état pathogène en fonction des valeurs différentes de délais

- Quelle propriété ?

Model-checking temporisé

La représentation sous UPPAAL des comportements chez *Pseudomonas aeruginosa*

Vérification de la propriété d'accessibilité de l'état pathogène en fonction des valeurs différentes de délais

- Quelle propriété ?
- Quelles variables sont-elles déterminantes ?

Model-checking temporisé

La représentation sous UPPAAL des comportements chez *Pseudomonas aeruginosa*

Vérification de la propriété d'accessibilité de l'état pathogène en fonction des valeurs différentes de délais

- Quelle propriété ?
- Quelles variables sont-elles déterminantes ?
- Quelles valeurs de ces variables pourront faire la différence ?

Model-checking temporisé

La représentation sous UPPAAL des comportements chez *Pseudomonas aeruginosa*

Vérification de la propriété d'accessibilité de l'état pathogène en fonction des valeurs différentes de délais

- Quelle propriété ? X_2 est accessible dans A ↪
- Quelles variables sont-elles déterminantes ?
- Quelles valeurs de ces variables pourront faire la différence ?

Model-checking temporisé

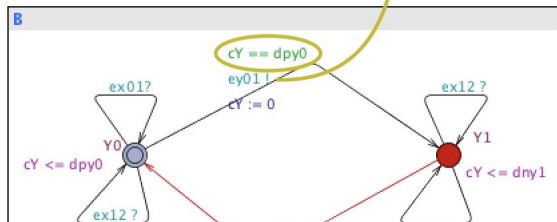
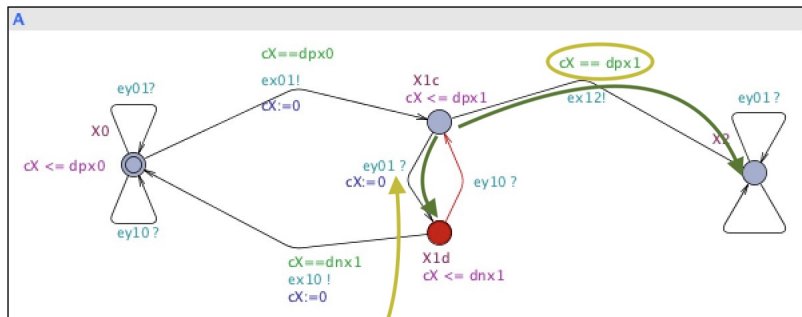
La représentation sous UPPAAL des comportements chez *Pseudomonas aeruginosa*

Vérification de la propriété d'accessibilité de l'état pathogène en fonction des valeurs différentes de délais

- Quelle propriété ? X_2 est accessible dans A \rightsquigarrow $E \ll X_2$
- Quelles variables sont-elles déterminantes ?
- Quelles valeurs de ces variables pourront faire la différence ?

Model-checking temporel

La représentation sous UPPAAL des comportements chez *Pseudomonas aeruginosa*



Model-checking temporisé

La représentation sous UPPAAL des comportements chez *Pseudomonas aeruginosa*

Vérification de la propriété d'accessibilité de l'état pathogène en fonction des valeurs différentes de délais

- Quelle propriété ? X_2 est accessible dans A $\rightsquigarrow E \langle \rangle A . X_2$
- Quelles variables sont-elles déterminantes ? $dp \times 1$ et $dp_y 0$
- Quelles valeurs de ces variables pourront faire la différence ?

Model-checking temporisé

La représentation sous UPPAAL des comportements chez *Pseudomonas aeruginosa*

Vérification de la propriété d'accessibilité de l'état pathogène en fonction des valeurs différentes de délais

- Quelle propriété ? X_2 est accessible dans A $\rightsquigarrow E \langle \langle A \rangle \rangle X_2$
- Quelles variables sont-elles déterminantes ? dp_{x1} et dp_{y0}
- Quelles valeurs de ces variables pourront faire la différence ?
 $dp_{y0} = 3$ et $\{ dp_{x1} = 7 \text{ ou } dp_{x1} = 2 \}$

Model-checking temporisé

La représentation sous UPPAAL des comportements chez *Pseudomonas aeruginosa*

Vérification de la propriété d'accessibilité de l'état pathogène en fonction des valeurs différentes de délais dp_{x1} et dp_{y0}

Status

(Academic) UPPAAL version 4.1.19 (rev. 5648), October 2014 -- server.
Déconnecté.

Connexion directe établi au serveur local.

(Academic) UPPAAL version 4.1.19 (rev. 5648), October 2014 -- server.
E<>A.X2

Vérification/noyau/temps total écoulé: 0s / 0.001s / 0.012s.

Pics utilisation de la mémoire permanente/virtuelle: {0}KB / {1}KB.

La propriété n'est pas satisfaite.

E<>A.X2

Vérification/noyau/temps total écoulé: 0.001s / 0s / 0.001s.

Pics utilisation de la mémoire permanente/virtuelle: {0}KB / {1}KB.

La propriété est satisfaite.

Figure 4 : $dp_{y0} = 3$ *et* $\{ dp_{x1} = 7$ *ou* $dp_{x1} = 2 \}$

Les outils pratiques pour la vérification des **automates temporisés** ou des automates hybrides (1)

▷ UPPAAL : <http://www.uppaal.org>

RELATED TOOLS: TIMES | Stratego | CORA | TRON | TIGA | SMC | COVER | PORT | PRO

UPPAAL

Home

Home | About | Documentation | Download | Examples | Web Help | Bugs

UPPAAL is an integrated tool environment for modeling, validation and verification of real-time systems modeled as networks of timed automata, extended with data types (bounded integers, arrays, etc.).

The tool is developed in collaboration between the [Department of Information Technology](#) at Uppsala University, Sweden and the [Department of Computer Science](#) at Aalborg University in Denmark.

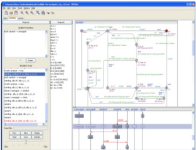


Figure 1: UPPAAL ON SCREEN.

License

The UPPAAL tool is free for non-commercial applications in academia **only**. For commercial applications a commercial license is required. Please see the [Download](#) section or www.uppaal.com for more information.

To find out more about UPPAAL, read this short [introduction](#). Further information may be found at this web site in the pages [About](#), [Documentation](#), [Download](#), and [Examples](#).

Mailing Lists

UPPAAL has an open [discussion forum](#) group at Yahoo!Groups intended for users of the

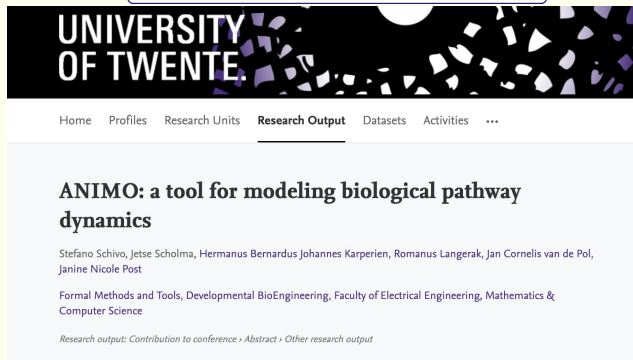
Download

News: The current official release is UPPAAL 4.0.14 (May 20, 2014). Compared to version 3, the 4.0 release is the result of over 6.5 years of additional development, and many new features and improvements are introduced (see also this [release note](#) and the web help section [new features](#)). To support models created in previous versions of UPPAAL, version 4.0 can convert most old models directly from the GUI (alternatively it can be run in 3.4 compatibility mode by defining the environment variable UPPAAL_OLD_SYNTAX, see also item 2 of the [FAQ](#)).

Les outils pratiques pour la vérification des **automates temporisés** ou des automates hybrides (1)

▷ UPPAAL : <http://www.uppaal.org>

▷ ANIMO : <https://fmt.ewi.utwente.nl/tools/webANIMO/>



UNIVERSITY OF TWENTE

Home Profiles Research Units **Research Output** Datasets Activities ...

ANIMO: a tool for modeling biological pathway dynamics

Stefano Schivo, Jetse Scholma, Hermanus Bernardus Johannes Karperien, Romanus Langerak, Jan Cornelis van de Pol, Janine Nicole Post

Formal Methods and Tools, Developmental BioEngineering, Faculty of Electrical Engineering, Mathematics & Computer Science

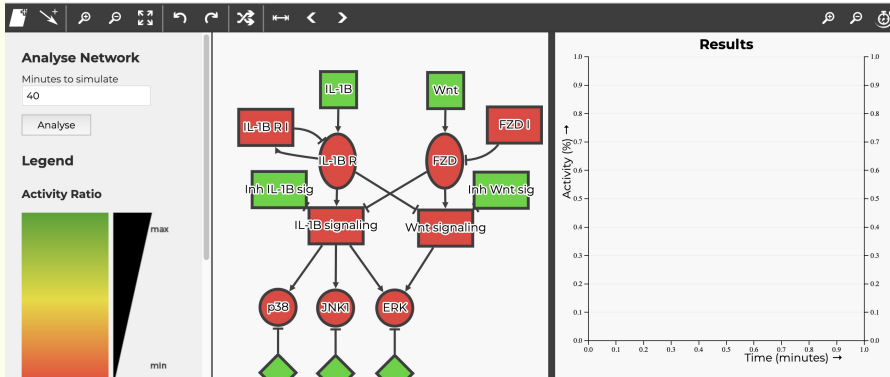
Research output: Contribution to conference › Abstract › Other research output

Les outils pratiques pour la vérification des **automates temporisés** ou des automates hybrides (1)

▷ UPPAAL : <http://www.uppaal.org>

▷ ANIMO : <https://fmt.ewi.utwente.nl/tools/webANIMO/>

webANIMO Network Examples Plot Help



Michiel Bakker, Jacco Brandt, Ruben Haasjes, Bob Rubbens, Willem Siers © 2016

For academic use only

Mais les automates temporisés **ne permettent pas** de bien prendre en compte :

- ▶ ni les différences de vitesses d'évolutions,
- ▶ ni les suspensions des activités (avec reprises ultérieures),
- ▶ ni les cumuls de ces évolutions successives.

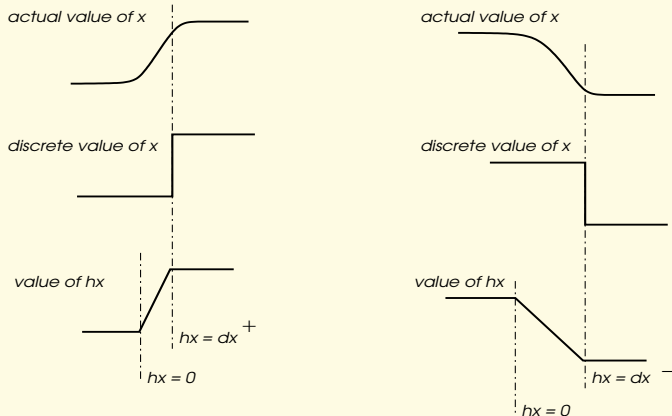
Sommaire

- 1 Introduction
- 2 Généralités — sur la représentation du temps dans les modèles de la dynamique des système
- 3 »»» PREMIERE PARTIE : Des modèles hybrides «««
- 4 Model-checking temporisé
- 5 Model-checking hybride**
- 6 Abstraction et raffinement
- 7 »»» DEUXIEME PARTIE : Des données aux modèles «««
- 8 Apprentissage des modèles
- 9 Extension hybride : du discret ... aux intervalles
- 10 Conclusion
- 11 Bibliographie

Modélisation par automates hybrides :

des horloges pour mesurer les délais de franchissement de seuil

Figure 5 : modélisation des délais par des “horloges”



Related works (suite)

■ Automates hybrides :

- ▶ [\[L. Bortolussi and A. Policriti\]*](#). "Hybrid Systems and Biology. Continuous and Discrete Modeling for Systems Biology". In *Formal Methods For Computational System Biology (FMCSB)*, LNCS 5016, 2006. [BP06]
- ▶ [\[P. Lincoln and A. Tiwari\]](#). "Symbolic systems biology : Hybrid modeling and analysis of biological networks". In *Hybrid Systems : Computation and Control (HSCC'04)*, LNCS 2993, 2004. [LT04]
- ▶ [\[G. Batt, C. Belta and R. Weiss\]](#). "Model checking liveness properties of genetic regulatory networks". In *Tools and algorithms for the construction and analysis of systems (TACAS'07)*, LNCS, 2007. [BBW07b]
- ▶ [\[G. Batt, C. Belta and R. Weiss\]](#). "Model checking genetic regulatory networks with parameter uncertainty". In *Hybrid systems (HSCC'07)*, LNCS, 2007. [BBW07a]
- ▶ [\[A. Aswani and C. Tomlin\]](#). "Reachability algorithm for biological piecewise-affine hybrid systems". In *Hybrid systems (HSCC'07)*, LNCS, 2007. [AT07]
- ▶ [\[J. Ahmad, G. Bernot, J.-P. Comet, D. Lime and O. Roux\]](#). "Hybrid modelling and dynamical analysis of gene regulatory networks with delays". *ComPlexUs*, 3(4) :231-251, 2006 (Cover Date : Nov. 2007)

EXAMPLES OF USE OF HS FOR SYSTEMS BIOLOGY

ESCHERICHIA COLI

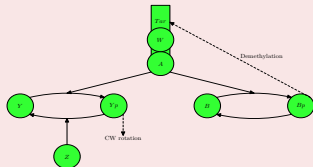
- a bacterium detecting the food concentration through a set of receptors;
- moving by flagellar rotations.



Depending on the concentration of attractants and repellents, *E. coli* responds to stimuli in one of two ways:

- “**RUNS**” – it moves in a straight line by moving its flagella counterclockwise (**CCW**)
- “**TUMBLES**” – it randomly changes its heading by moving its flagella clockwise (**CW**)

E. Coli MODEL



EXAMPLES OF USE OF HS FOR SYSTEMS BIOLOGY

E. coli IDA MODEL

RUN [CCW]

$$\omega = -1$$

$$\dot{Y}_P = k_y P(Y_0 - Y_P) - k_{-y} Z Y_P$$

$$\dot{B}_P = k_b P(B_0 - B_P) - k_{-b} B_P$$

$$P = LT_{2p} + LT_{3p} + LT_{4p} + T_{2p} + T_{3p} + T_{4p}$$

$$y = \frac{Y_P}{Y_0} > \theta \wedge \omega' = +1 \wedge Y'_P = Y_P \wedge Y'_0 = Y_0 \wedge B'_P = B_P \wedge B'_0 = B_0 \wedge Z' = Z \wedge P' = P$$

TUMBLE [CW]

$$\omega = +1$$

$$\dot{Y}_P = k_y P(Y_0 - Y_P) - k_{-y} Z Y_P$$

$$\dot{B}_P = k_b P(B_0 - B_P) - k_{-b} B_P$$

$$P = LT_{2p} + LT_{3p} + LT_{4p} + T_{2p} + T_{3p} + T_{4p}$$

$$y = \frac{Y_P}{Y_0} < \theta \wedge \omega' = -1 \wedge Y'_P = Y_P \wedge Y'_0 = Y_0 \wedge B'_P = B_P \wedge B'_0 = B_0 \wedge Z' = Z \wedge P' = P$$

A. Casagrande et al., *Independent Dynamics Hybrid Automata in Systems Biology*, AB('05) Tokyo, 2005

Automates hybrides (suite)

Une première forme de la définition

- Un ensemble fini de **localités** l
- Un ensemble fini de **variables** v dans \mathbb{R}
- Un ensemble fini d'**états initiaux** (couples (l, v))
- Un ensemble fini de **transitions**
- Une **fonction de flux**, qui associe à chaque localité un prédicat sur \dot{v}
- Une **fonction d'invariant**, qui associe à chaque localité un prédicat sur v
- Une **fonction de condition de saut**, qui associe à chaque localité un prédicat sur v
- Une **condition d'initialisation**, qui associe à l'état initial un prédicat
- *Un ensemble fini d'étiquettes de synchronisation*

Automates hybrides linéaires [Hen96]

Les idées clefs

- Les fonctions d'invariants, de flux et de saut sont des **combinaisons booléennes d'égalités linéaires**.
- À chaque localité est associé **un ensemble d'équations différentielles ordinaires** $\sum \dot{x} \leq k$, où $k \in \mathbb{R}$, définissant l'évolution des variables.

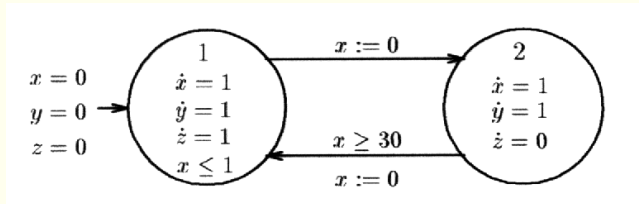


Figure extraite de : [Hen96]

Introduction aux automates hybrides par un exemple en biologie :

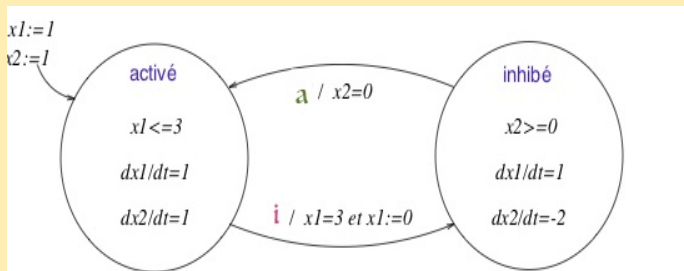
évolution du niveau d'expression d'un gène qui subit deux influences opposées

- inhibition toutes les 3 unités de temps jusqu'à ce qu'il atteigne le niveau basal puis activation
 - localité 1 (en croissance) : à la vitesse 1
 - localité 2 (en décroissance) : à la vitesse 2
-
- deux variables : x_1 : le temps ; et x_2 : le niveau d'expression

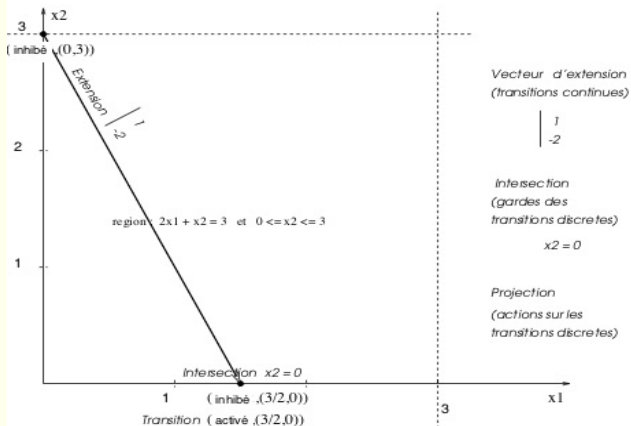
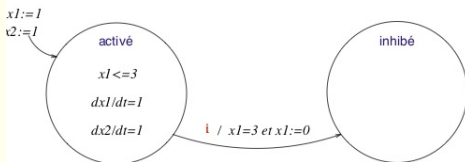
Introduction aux automates hybrides par un exemple en biologie : évolution du niveau d'expression d'un gène qui subit deux influences opposées

- inhibition toutes les 3 unités de temps jusqu'à ce qu'il atteigne le niveau basal puis activation
 - localité 1 (en croissance) : à la vitesse 1
 - localité 2 (en décroissance) : à la vitesse 2
- deux variables : x_1 : le temps ; et x_2 : le niveau d'expression

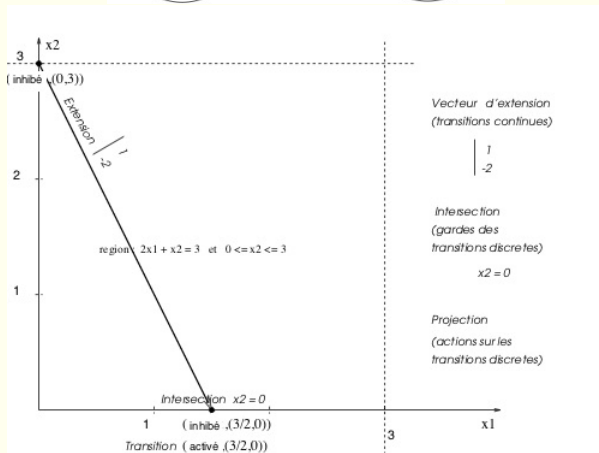
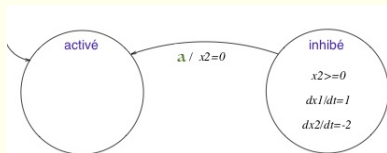
Automate hybride représentant le niveau d'expression du gène



Activation



Inhibition



Définition [ACH+95] :

Rappel de la définition des automates temporisés

Automate hybride H : 7-uplet $(Loc, Var, T, E, Act, Inv, Lo)$ où :

- 1 Loc : ensemble fini de **localités**,
- 2 Var : ensemble fini de **variables** réelles,
- 3 E : ensemble fini d'**étiquettes**,
- 4 T : ensemble fini de **transitions** discrètes $(l', \gamma, e, \alpha, l) \in T$,
- 5 Act : fonction associant à chaque localité $l \in Loc$ un ensemble d'**activités** $Act(l)$
- 6 Inv : fonction associant à chaque localité $l \in Loc$ un **invariant** $Inv(l)$.
- 7 $Lo \subset Loc$: ensemble des **localités initiales**,

Définition [ACH+95] :

Rappel de la définition des automates temporisés

Automate hybride H : 7-uplet $(Loc, Var, T, E, Act, Inv, Lo)$ où :

- 1 Loc : ensemble fini de **localités**,
- 2 Var : ensemble fini de **variables** réelles,
- 3 E : ensemble fini d'**étiquettes**,
- 4 T : ensemble fini de **transitions** discrètes $(l', \gamma, e, \alpha, l) \in T$,
- 5 Act : fonction associant à chaque localité $l \in Loc$ un ensemble d'**activités** $Act(l)$
- 6 Inv : fonction associant à chaque localité $l \in Loc$ un **invariant** $Inv(l)$.
- 7 $Lo \subset Loc$: ensemble des **localités initiales**,

▷ **Valuation** v :

$$v : \begin{array}{ll} Var & \rightarrow R \\ x & \mapsto v(x) \end{array}$$

\mathcal{V} : ensemble des valuations

- ▷ **Région symbolique** : domaine regroupant un ensemble de points, valuations de toutes les variables.
- ▷ **Exécution** : suite alternante de transitions **continues** et **discrètes**.

Modélisation par automates hybrides : (suite)

Semi-algorithmes d'accessibilité :

Problème d'accessibilité : déterminer si un domaine final F est accessible à partir d'un domaine initial I .

Deux façons :

- Considérer le domaine $succ^*$ des états qui peuvent être atteints à partir du domaine initial I ,
et vérifier si $succ^* \cap F \neq \emptyset$
(résolution « en avant »)
- Considérer le domaine $prec^*$ des états à partir desquels les états de F peuvent être atteints,
et vérifier si $prec^* \cap I \neq \emptyset$
(résolution « en arrière »).

Modélisation par automates hybrides : (suite)

Semi-algorithme d'accessibilité « en avant »

Domaine $succ^*$: union des domaines $\bigcup_{k \geq 0} succ_k$, $succ_k$ définis

itérativement par :

$$succ_0 = \bigcup_{I \in Loc} (I, (I(I) \nearrow Act(I)) \cap Inv(I)),$$

$$succ_{i+1} = \bigcup_{I \in Loc} (I, \bigcup_{(I', \gamma, \alpha, l) \in Tra} ((\alpha(succ_i(I')) \cap \gamma) \nearrow Act(I)) \cap Inv(I)).$$

$$succ^n = \bigcup_{k=0..n} succ_k.$$

Modélisation par automates hybrides : (suite)

Semi-algorithme d'accessibilité « en avant »

Domaine $succ^*$: union des domaines $\bigcup_{k \geq 0} succ_k$, $succ_k$ définis itérativement par :

$$succ_0 = \bigcup_{I \in Loc} (I, (I(I) \nearrow Act(I)) \cap Inv(I)),$$

$$succ_{i+1} = \bigcup_{I \in Loc} (I, \bigcup_{(I', \gamma, \alpha, I') \in Tra} ((\alpha(succ_i(I')) \cap \gamma) \nearrow Act(I)) \cap Inv(I)).$$

$$succ^n = \bigcup_{k=0..n} succ_k.$$

Semi-algorithme d'accessibilité « en arrière »

Domaine $prec^*$: union des domaines $\bigcup_{k \geq 0} prec_k$, $prec_k$ définis itérativement par :

$$prec_0 = \bigcup_{I \in Loc} (I, (F(I) \swarrow Act(I)) \cap Inv(I)),$$

$$prec_{i+1} = \bigcup_{I \in Loc} (I, \bigcup_{(I', \gamma, \alpha, I') \in Tra} ((\alpha^{-1}(prec_i(I')) \cap \gamma) \swarrow Act(I)) \cap Inv(I)).$$

$$prec^n = \bigcup_{k=0..n} prec_k.$$

Modélisation par automates hybrides : (suite)

Terminaison des algorithmes :

- soit lorsque $succ^{n+1} = succ^n$ ou $prec^{n+1} = prec^n$ (point fixe),
- soit lorsque $succ^n \cap F \neq \emptyset$ ou $prec^n \cap I \neq \emptyset$
(réponse au problème : « oui, le domaine final F peut être atteint »).

Modélisation par automates hybrides : (suite)

Terminaison des algorithmes :

- soit lorsque $succ^{n+1} = succ^n$ ou $prec^{n+1} = prec^n$ (point fixe),
- soit lorsque $succ^n \cap F \neq \emptyset$ ou $prec^n \cap I \neq \emptyset$
(réponse au problème : « oui, le domaine final F peut être atteint »).

↪ *Cependant, rien ne permet d'affirmer que l'une au moins de ces conditions sera vérifiée à une itération k : problème indécidable.*

↪ c'est l'inconvénient des automates hybrides...

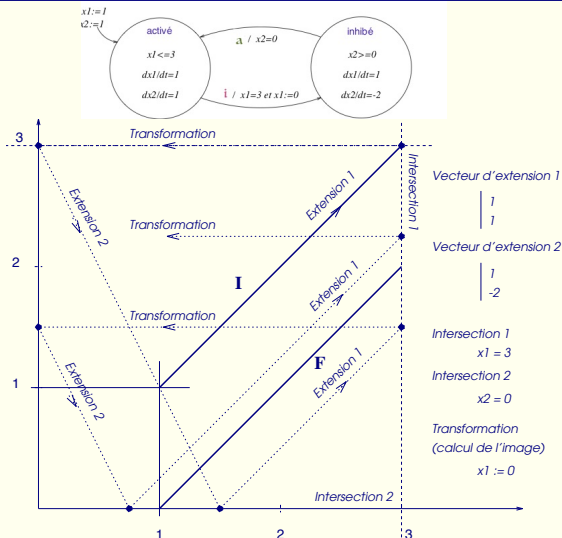
Modélisation par automates hybrides : (suite)

Retour sur l'exemple : recherche de trajectoire

$$I = x1 = x2 \wedge 1 \leq x1 \leq 3$$

$$F = x1 = x2 + 1 \wedge 1 \leq x1 \leq 3$$

en avant

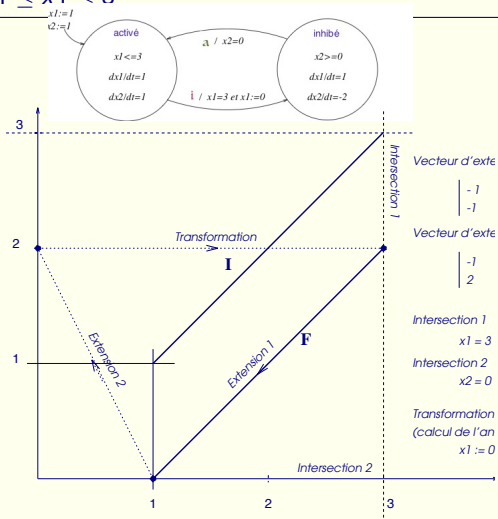


Modélisation par automates hybrides : (suite)

$$I = x1 = x2 \wedge 1 \leq x1 \leq 3$$

$$F = x1 = x2 + 1 \wedge 1 \leq x1 < 3$$

en arrière

Figure 8 : F n'est pas accessible !

Modélisation par automates hybrides : (suite et récapitulation)

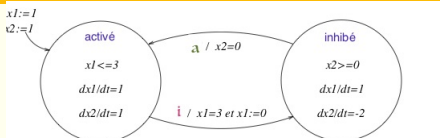


Figure 10 : Activation et inhibition d'un gène

- Région initiale : $I = (\text{activé}, x_1 = x_2 \wedge x_1 \leq 3 \wedge x_2 \geq 0)$
- Région finale : $F = (\text{inhibé}, 1 \leq x_1 \leq 3 \wedge x_1 = x_2 + 1)$

- en arrière :

$$prec^* = (\text{activé}, 1 \leq x_1 \leq 3 \wedge x_1 = x_2 + 1) \vee (\text{inhibé}, x_1 + 2x_2 = 2)$$

et $prec^* \cap I = \emptyset$

- en avant :

$$succ_{2i} = (\text{activé}, x_1 = x_2 + 1 + \frac{(-1)^{i+1}}{2^i} \wedge x_1 \leq 3) \vee (\text{inhibé}, \emptyset)$$

$$succ_{2i+1} = (\text{activé}, \emptyset) \vee (\text{inhibé}, 2x_1 + x_2 = 2 + \frac{(-1)^i}{2^i} \wedge x_2 \geq 0)$$

→ non convergence

Application de la modélisation hybride pour obtenir des résultats sur l'évolution dans *Pseudomonas*. [BMC Syst Biol. 2010 ; 4 : 79]

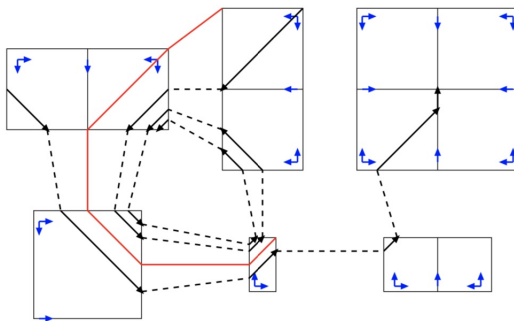


FIG. 5 – Dynamique hybride avec spirale convergente.

Application de la modélisation hybride pour obtenir des résultats sur l'évolution dans Pseudomonas. [BMC Syst Biol. 2010 ; 4 : 79]

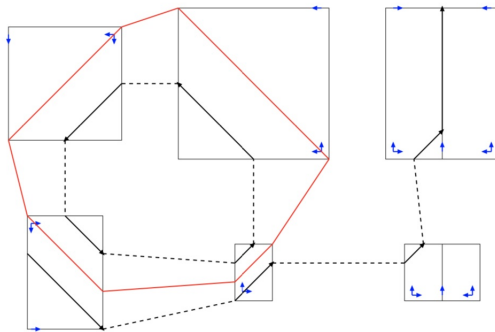


FIG. 6 – Dynamique hybride avec cycle.

Application de la modélisation hybride pour obtenir des résultats sur l'évolution dans *Pseudomonas*. [BMC Syst Biol. 2010 ; 4 : 79]

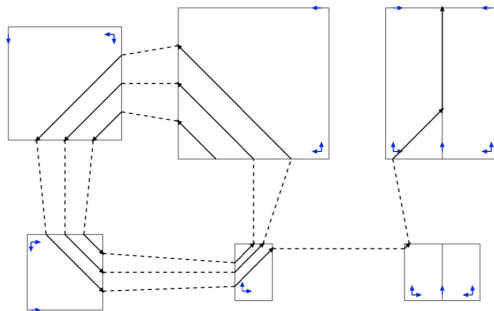
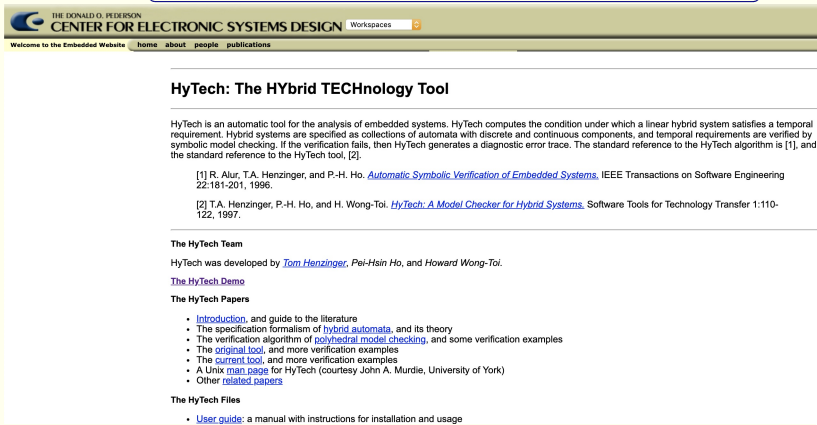


FIG. 7 – Dynamique hybride avec spirale divergente.

Les outils pratiques pour la vérification des automates temporisés ou des **automates hybrides** (2)

▷ HYTECH : <https://ptolemy.berkeley.edu/projects/embedded/research/hytech>



THE DONALD G. PEDERSON
CENTER FOR ELECTRONIC SYSTEMS DESIGN Workspaces

Welcome to the Embedded Website home about people publications

HyTech: The HYbrid TECHNOlogy Tool

HyTech is an automatic tool for the analysis of embedded systems. HyTech computes the condition under which a linear hybrid system satisfies a temporal requirement. Hybrid systems are specified as collections of automata with discrete and continuous components, and temporal requirements are verified by symbolic model checking. If the verification fails, then HyTech generates a diagnostic error trace. The standard reference to the HyTech algorithm is [1], and the standard reference to the HyTech tool, [2].

[1] R. Alur, T.A. Henzinger, and P.-H. Ho. *Automatic Symbolic Verification of Embedded Systems*, IEEE Transactions on Software Engineering 22:181-201, 1996.

[2] T.A. Henzinger, P.-H. Ho, and H. Wong-Toi. *HyTech: A Model Checker for Hybrid Systems*. Software Tools for Technology Transfer 1:110-122, 1997.

The HyTech Team

HyTech was developed by [Tom Henzinger](#), [Pei-Hsin Ho](#), and [Howard Wong-Toi](#).

[The HyTech Demo](#)

The HyTech Papers

- [Introduction](#), and guide to the literature
- The specification formalism of [hybrid automata](#), and its theory
- The verification algorithm of [polyhedral model checking](#), and some verification examples
- The [original tool](#), and more verification examples
- The [current tool](#), and more verification examples
- A Unix [man page](#) for HyTech (courtesy John A. Murdie, University of York)
- Other [related papers](#)

The HyTech Files

- [User guide](#): a manual with instructions for installation and usage

Les outils pratiques pour la vérification des automates temporisés ou des **automates hybrides** (2)

▶ **HYTECH** : <https://ptolemy.berkeley.edu/projects/embedded/research/hytech>

▶ **SPACEEX** : <http://www.spaceex.imag.fr/>

SpaceEx State Space Explorer

Home

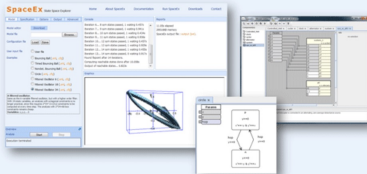
About SpaceEx

Documentation

Run SpaceEx

Downloads

Contact



- Learn more about SpaceEx
- Download SpaceEx
- Subscribe to the newsletter

spaceex.imag.fr

The verification of continuous and hybrid systems is a challenging problem, and various approaches are currently being investigated to overcome the complexities of representing and computing with continuous sets of states. Since verification problems are generally undecidable for such systems, experimental results are vital for evaluating and developing new ideas.

The SpaceEx tool platform is designed to facilitate the implementation of algorithms related to reachability and safety verification.

Latest News

Examples and Tutorials

Sommaire

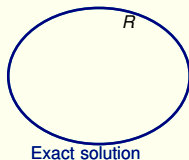
- 1 Introduction
- 2 Généralités — sur la représentation du temps dans les modèles de la dynamique des système
- 3 »»» PREMIERE PARTIE : Des modèles hybrides «««
- 4 Model-checking temporisé
- 5 Model-checking hybride
- 6 Abstraction et raffinement**
- 7 »»» DEUXIEME PARTIE : Des données aux modèles «««
- 8 Apprentissage des modèles
- 9 Extension hybride : du discret ... aux intervalles
- 10 Conclusion
- 11 Bibliographie

Abstractions pour l'analyse d'accessibilité (par exemple)

Vérification d'une propriété d'accessibilité \mathcal{R} (pour *Reach*) :

$\mathcal{R} \equiv$ « Depuis un état s_0 , est-il possible d'atteindre un état s_n donné ? »

Approximations : \mathcal{P} et \mathcal{Q} , construits de sorte que $\mathcal{P} \Rightarrow \mathcal{R} \Rightarrow \mathcal{Q}$

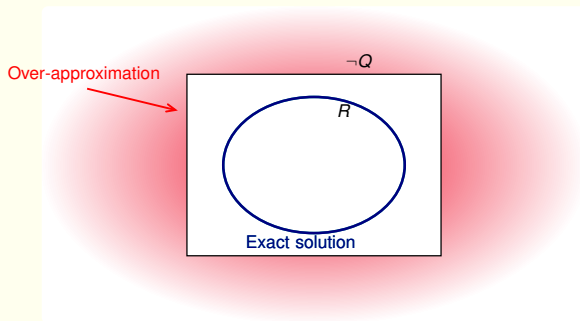


Abstractions pour l'analyse d'accessibilité (par exemple)

Vérification d'une propriété d'accessibilité \mathcal{R} (pour *Reach*) :

$\mathcal{R} \equiv$ « Depuis un état s_0 , est-il possible d'atteindre un état s_n donné ? »

Approximations : \mathcal{P} et \mathcal{Q} , construits de sorte que $\mathcal{P} \Rightarrow \mathcal{R} \Rightarrow \mathcal{Q}$

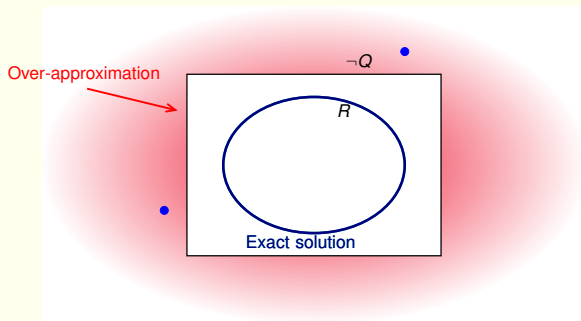


Abstractions pour l'analyse d'accessibilité (par exemple)

Vérification d'une propriété d'accessibilité \mathcal{R} (pour *Reach*) :

$\mathcal{R} \equiv$ « Depuis un état s_0 , est-il possible d'atteindre un état s_n donné ? »

Approximations : \mathcal{P} et \mathcal{Q} , construits de sorte que $\mathcal{P} \Rightarrow \mathcal{R} \Rightarrow \mathcal{Q}$

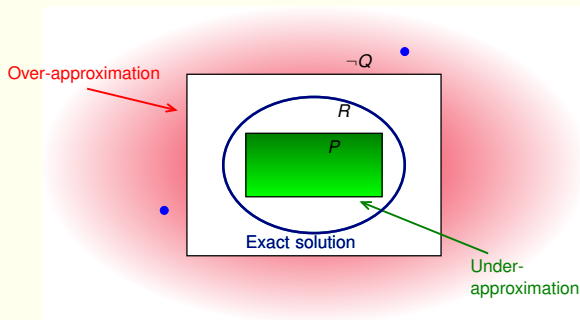


Abstractions pour l'analyse d'accessibilité (par exemple)

Vérification d'une propriété d'accessibilité \mathcal{R} (pour *Reach*) :

$\mathcal{R} \equiv$ « Depuis un état s_0 , est-il possible d'atteindre un état s_n donné ? »

Approximations : \mathcal{P} et \mathcal{Q} , construits de sorte que $\mathcal{P} \Rightarrow \mathcal{R} \Rightarrow \mathcal{Q}$

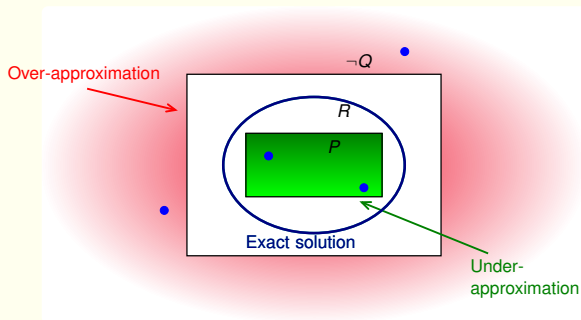


Abstractions pour l'analyse d'accessibilité (par exemple)

Vérification d'une propriété d'accessibilité \mathcal{R} (pour *Reach*) :

$\mathcal{R} \equiv$ « Depuis un état s_0 , est-il possible d'atteindre un état s_n donné ? »

Approximations : \mathcal{P} et \mathcal{Q} , construits de sorte que $\mathcal{P} \Rightarrow \mathcal{R} \Rightarrow \mathcal{Q}$

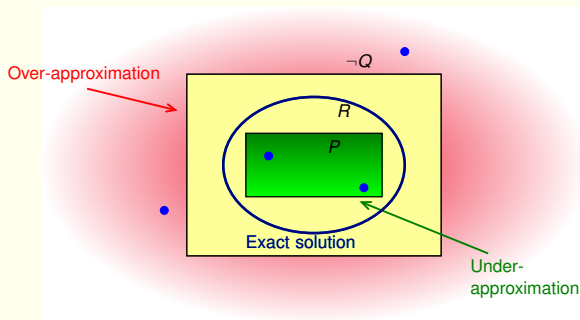


Abstractions pour l'analyse d'accessibilité (par exemple)

Vérification d'une propriété d'accessibilité \mathcal{R} (pour *Reach*) :

$\mathcal{R} \equiv$ « Depuis un état s_0 , est-il possible d'atteindre un état s_n donné ? »

Approximations : \mathcal{P} et \mathcal{Q} , construits de sorte que $\mathcal{P} \Rightarrow \mathcal{R} \Rightarrow \mathcal{Q}$

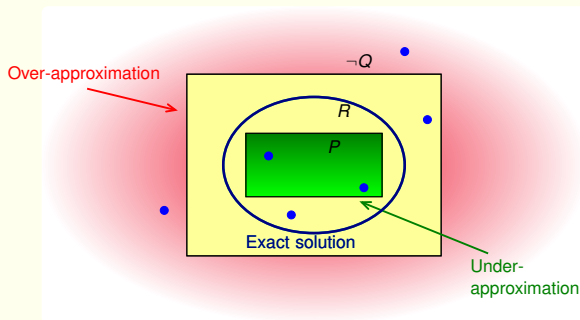


Abstractions pour l'analyse d'accessibilité (par exemple)

Vérification d'une propriété d'accessibilité \mathcal{R} (pour *Reach*) :

$\mathcal{R} \equiv$ « Depuis un état s_0 , est-il possible d'atteindre un état s_n donné ? »

Approximations : \mathcal{P} et \mathcal{Q} , construits de sorte que $\mathcal{P} \Rightarrow \mathcal{R} \Rightarrow \mathcal{Q}$

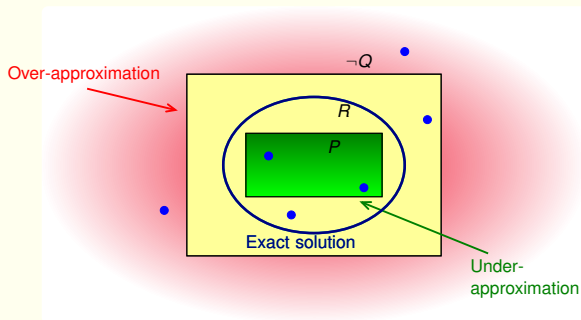


Abstractions pour l'analyse d'accessibilité (par exemple)

Vérification d'une propriété d'accessibilité \mathcal{R} (pour *Reach*) :

$\mathcal{R} \equiv$ « Depuis un état s_0 , est-il possible d'atteindre un état s_n donné ? »

Approximations : \mathcal{P} et \mathcal{Q} , construits de sorte que $\mathcal{P} \Rightarrow \mathcal{R} \Rightarrow \mathcal{Q}$



\rightsquigarrow Zone inconclusive

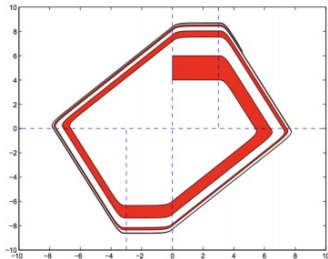
Abstractions

- ▶ Calcul de zones englobantes (polygones englobants, enveloppe convexe, ...)
... sur-approximation des états accessibles

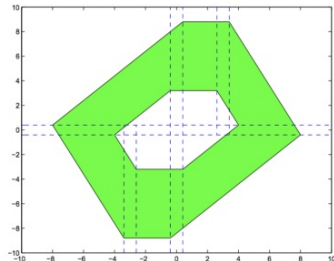
Abstractions

▷ Calcul de zones englobantes (polygones englobants, enveloppe convexe, ...)

... sur-approximation des états accessibles
 ... par exemple dans SPACEEX :



(a) H_1 : piecewise affine dynamics,
6 variables



(b) H_2 : pw. constant dynamics,
2 variables, $H_1 \succeq_{0.4} H_2$

reachable states much easier to compute for H_2

Abstractions

- ▶ Calcul de zones englobantes (polygones englobants, enveloppe convexe, ...)
... sur-approximation des états accessibles

- ▶ Maintien de la chronologie et abstraction de la chronométrie
... ce qui se fait couramment

Abstractions

- ▶ Calcul de zones englobantes (polygones englobants, enveloppe convexe, ...)
 - ... sur-approximation des états accessibles

- ▶ Maintien de la chronologie et abstraction de la chronométrie
 - ... ce qui se fait couramment

- ▶ Abstraction de la chronologie et raffinement par la chronométrie
 - ... une démarche originale qui peut éviter de construire le graphe complet... \rightsquigarrow réseaux d'automates asynchrones ((PINT))

Abstractions

- ▶ Calcul de zones englobantes (polygones englobants, enveloppe convexe, ...)
 - ... sur-approximation des états accessibles

- ▶ Maintien de la chronologie et abstraction de la chronométrie
 - ... ce qui se fait couramment

- ▶ Abstraction de la chronologie et raffinement par la chronométrie
 - ... une démarche originale qui peut éviter de construire le graphe complet... \rightsquigarrow réseaux d'automates asynchrones ((PINT))

- ▶ ...



Sommaire

- 1 Introduction
- 2 Généralités — sur la représentation du temps dans les modèles de la dynamique des système
- 3 »»» PREMIERE PARTIE : Des modèles hybrides «««
- 4 Model-checking temporisé
- 5 Model-checking hybride
- 6 Abstraction et raffinement
- 7 »»» DEUXIEME PARTIE : Des données aux modèles «««
- 8 Apprentissage des modèles
- 9 Extension hybride : du discret ... aux intervalles
- 10 Conclusion
- 11 Bibliographie

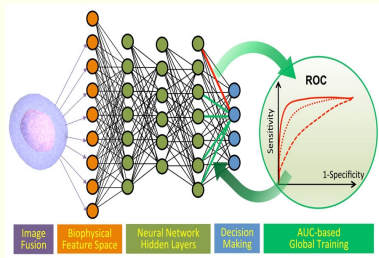
Sommaire

- 1 Introduction
- 2 Généralités — sur la représentation du temps dans les modèles de la dynamique des système
- 3 »»» PREMIERE PARTIE : Des modèles hybrides «««
- 4 Model-checking temporisé
- 5 Model-checking hybride
- 6 Abstraction et raffinement
- 7 »»» DEUXIEME PARTIE : Des données aux modèles «««
- 8 Apprentissage des modèles**
- 9 Extension hybride : du discret ... aux intervalles
- 10 Conclusion
- 11 Bibliographie

Apprentissage des modèles

Diagnostic (*diagnosis*)

- ▶ Du grec ancien dia ("à travers") et gnostique
- ▶ A travers la connaissance

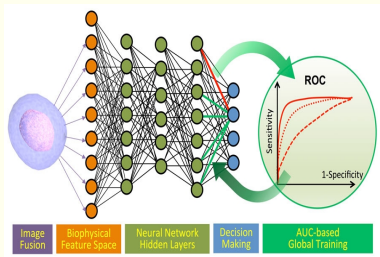


- ▶ ce qu'on comprend souvent par "apprentissage automatique" : reconnaître ce qui est
- ▶ Applications à la reconnaissance d'image

Apprentissage des modèles

Diagnostic (*diagnosis*)

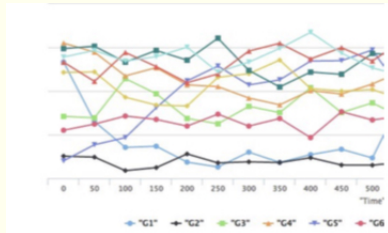
- ▷ Du grec ancien dia ("à travers") et gnostique
- ▷ A travers la connaissance



- ▷ ce qu'on comprend souvent par "apprentissage automatique" : reconnaître ce qui est
- ▷ Applications à la reconnaissance d'image

Pronostic (*prognosis*)

- ▷ Du grec ancien pro et gnostique
- ▷ Qui précède la connaissance, qui est produit avant.

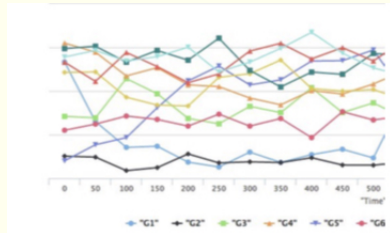


- ▷ ce que nous faisons en "apprentissage automatique" : connaître ce qui sera
- ▷ Applications à la compréhension du fonctionnement
 - ↪ anticipation des fonctionnements à venir

Apprentissage des modèles

Pronostic (*prognosis*)

- ▷ Du grec ancien pro et gnostique
- ▷ Qui précède la connaissance, qui est produit avant.



- ▷ ce que nous faisons en "apprentissage automatique" : **connaître ce qui sera**
- ▷ Applications à la compréhension du fonctionnement
 - ↪ anticipation des fonctionnements à venir

Interprétation des données de séries temporelles

*« S'il pleut à la Saint-Médard,
Il pleut quarante jours plus tard... »*

Interprétation des données de séries temporelles

*« S'il pleut à la Saint-Médard,
Il pleut quarante jours plus tard... »*

*« à moins que Saint-Barnabé ne lui
coupe l'herbe sous le pied. »*

Interprétation des données de séries temporelles

*« S'il pleut à la Saint-Médard,
Il pleut quarante jours plus tard... »*

*« à moins que Saint-Barnabé ne lui
coupe l'herbe sous le pied. »*

$T := \text{St_Medard} + 40$

$\text{pluie}(T) :- \text{pluie}(T - 40)$
 $\neg \text{pluie}(T) :- \text{soleil}(T - 37)$

Modeling gene interactions

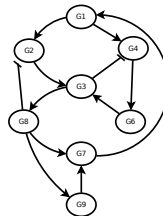
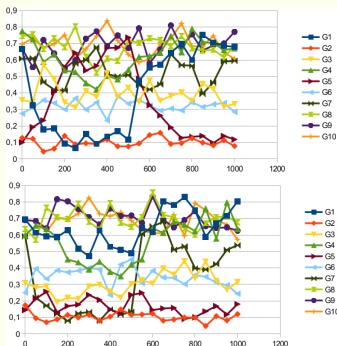
Goal : understand biological dynamics, i.e. gene interactions.

Data : time series

- discrete/regular time steps
- continuous values

Model : Boolean network

- discrete/regular time steps
- discrete values



Modeling gene interactions

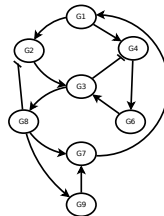
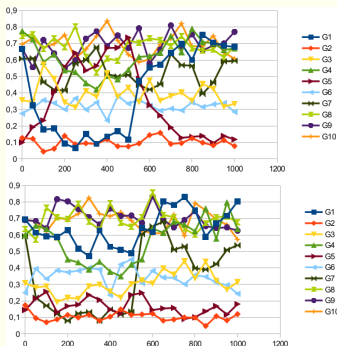
Goal : understand biological dynamics, i.e. gene interactions.

Data : time series

- discrete/regular time steps
- continuous values

Model : Boolean network

- discrete/regular time steps
- discrete values



<https://www.frontiersin.org/articles/10.3389/fbioe.2014.00081/full>

Learning delayed influences of biological systems

L'apprentissage des influences « retardées »
dans les systèmes biologiques

<https://www.frontiersin.org/articles/10.3389/fbioe.2014.00081/full>

Learning delayed influences of biological systems



SECTION ABOUT ARTICLES RESEARCH TOPICS FOR AUTHORS EDITORIAL BOARD



ARTICLE ALERTS

< Articles

THIS ARTICLE IS PART OF THE RESEARCH TOPIC

Computational methods for understanding complexity: the use of formal methods in biology

ORIGINAL RESEARCH ARTICLE

Front. Bioeng. Biotechnol., 16 January 2015 | <https://doi.org/10.3389/fbioe.2014.00081>

Learning delayed influences of biological systems

Tony Ribeiro¹, Morgan Magnin^{2,3}, Katsumi Inoue^{1,2} and Chiaki Sakama⁴

Capture d'écran



Download
Article



Export
citation

4,074

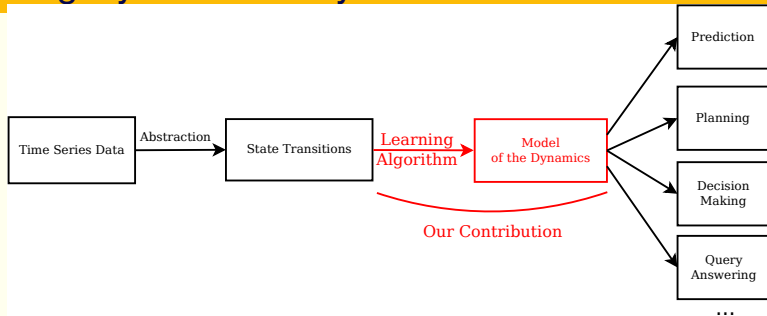
TOTAL VIEWS



1



Learning Dynamics of Systems from State Transitions



Challenges

From time series data, get a *relevant/efficient* abstraction of the system.

Goal

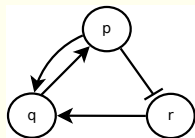
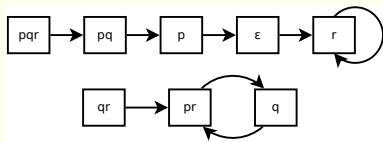
Automated modeling of **systems dynamics** from these data.

Learning From Interpretation Transitions (LFIT) (Inoue et al. 2014)

A framework for learning system dynamics from state transitions.

■ Basic Idea :

- Learn a logic program by observing the behavior of a system.
- This logic program captures the dynamics of the system.



— **Input** : Behavior of the system

—

— **Output** : Dynamics of the system

$$\begin{aligned}
 p(t+1) &\leftarrow q(t). \\
 q(t+1) &\leftarrow p(t) \wedge r(t). \\
 r(t+1) &\leftarrow \neg p(t).
 \end{aligned}$$

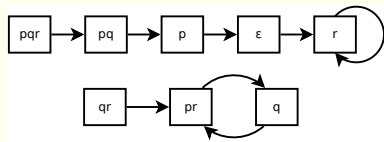
— **Representation** : Logic Program

Learning From Interpretation Transitions (LFIT) (Inoue et al. 2014)

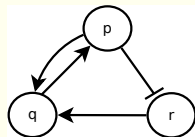
A framework for learning system dynamics from state transitions.

■ Basic Idea :

- Learn a logic program by observing the behavior of a system.
- This logic program captures the dynamics of the system.



— **What** happens



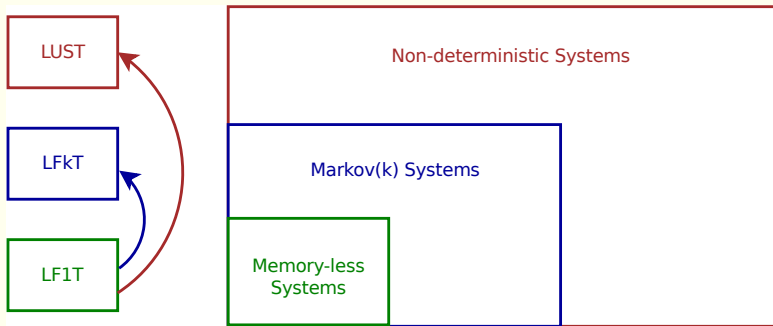
— **When** does it happen

$$\begin{aligned}
 p(t+1) &\leftarrow q(t). \\
 q(t+1) &\leftarrow p(t) \wedge r(t). \\
 r(t+1) &\leftarrow \neg p(t).
 \end{aligned}$$

— **Representation** : Logic Program

LFIT Algorithms

Three algorithms for three types of dynamics.



Learning from 1-step transition (LF1T) *(Inoue et al. MLJ 2013)*

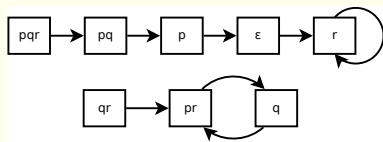
Input :

- The variables of the systems (pqr)
- A set E of state transitions (trace of executions)

Output :

- A set of rules P which represents the dynamics of the system

State Transitions



LF1T Input

$$E = \{ (pqr, pq), (pq, p), (p, \emptyset), (\emptyset, r), (r, r), (qr, pr), (pr, q), (q, pr) \}$$

Learning from 1 step transition *(Inoue et al. MLJ 2013)*

Let $(I, J) \in E$, for each $A \in J$ we can infer a **positive** rule R_A^I :

$$R_A^I := (A \leftarrow \bigwedge_{B_i \in I} B_i \wedge \bigwedge_{C_j \in \beta \setminus I} \neg C_j)$$

Exemple

From the state transition (pqr, pq) we can infer 2 rules :

- $p \leftarrow p \wedge q \wedge r.$
- $q \leftarrow p \wedge q \wedge r.$

Learning from 1 step transition *(Inoue et al. MLJ 2013)*

Let $(I, J) \in E$, for each $A \in J$ we can infer a **positive** rule R_A^I :

$$R_A^I := (A \leftarrow \bigwedge_{B_i \in I} B_i \wedge \bigwedge_{C_j \in \beta \setminus I} \neg C_j)$$

Exemple

From the next transition (pq, p) we can infer 1 logic rule :

- $p \leftarrow p \wedge q \wedge \neg r.$

Generalization Techniques

Generalize knowledge to learn the real relationship.

- $p \leftarrow p \wedge q \wedge r.$
- $p \leftarrow p \wedge q \wedge \neg r.$

Generalization Techniques

Generalize knowledge to learn the real relationship.

- $p \leftarrow p \wedge q \wedge r.$
- $p \leftarrow p \wedge q \wedge \neg r.$
- $p \leftarrow p \wedge q.$

Generalization Techniques

Generalize knowledge to learn the real relationship.

- $p \leftarrow p \wedge q.$
- $p \leftarrow \neg p \wedge q.$

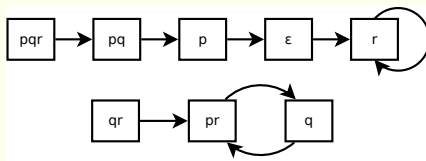
Generalization Techniques

Generalize knowledge to learn the real relationship.

- $p \leftarrow p \wedge q.$
- $p \leftarrow \neg p \wedge q.$
- $p \leftarrow q.$

Running Example

Input State Transitions



Learned Rules :

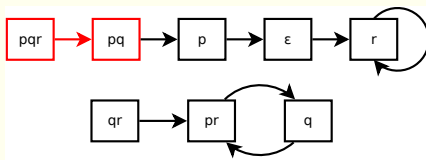
\emptyset

Knowledge Base :

\emptyset

Running Example

Input State Transitions



Learned Rules :

$$p \leftarrow p \wedge q \wedge r.$$

$$q \leftarrow p \wedge q \wedge r.$$

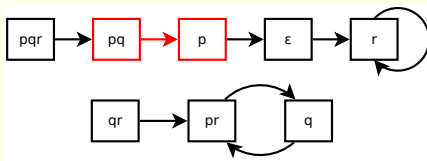
Knowledge Base :

$$p \leftarrow p \wedge q \wedge r.$$

$$q \leftarrow p \wedge q \wedge r.$$

Running Example

Input State Transitions

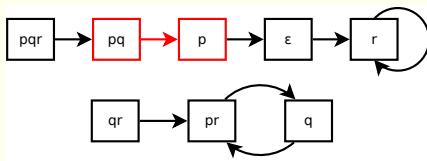


Learned Rules :
 $p \leftarrow p \wedge q \wedge \neg r.$

Knowledge Base :
 $p \leftarrow p \wedge q \wedge r.$
 $q \leftarrow p \wedge q \wedge r.$

Running Example

Input State Transitions

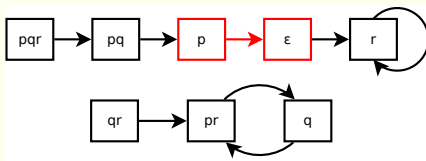


Learned Rules :
 $p \leftarrow p \wedge q \wedge \neg r.$

Knowledge Base :
 $p \leftarrow p \wedge q.$
 $q \leftarrow p \wedge q \wedge r.$

Running Example

Input State Transitions



Learned Rules :

\emptyset

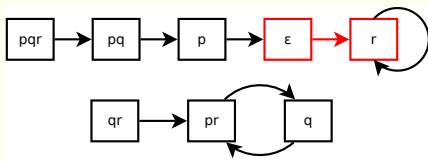
Knowledge Base :

$p \leftarrow p \wedge q.$

$q \leftarrow p \wedge q \wedge r.$

Running Example

Input State Transitions



Learned Rules :

$$r \leftarrow \neg p \wedge \neg q \wedge \neg r.$$

Knowledge Base :

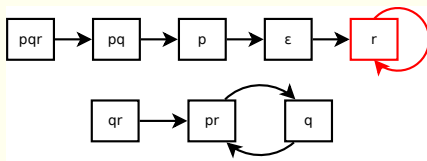
$$p \leftarrow p \wedge q.$$

$$q \leftarrow p \wedge q \wedge r.$$

$$r \leftarrow \neg p \wedge \neg q \wedge \neg r.$$

Running Example

Input State Transitions

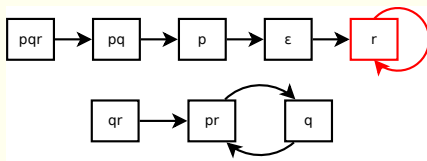


Learned Rules :
 $r \leftarrow \neg p \wedge \neg q \wedge r.$

Knowledge Base :
 $p \leftarrow p \wedge q.$
 $q \leftarrow p \wedge q \wedge r.$
 $r \leftarrow \neg p \wedge \neg q \wedge \neg r.$

Running Example

Input State Transitions

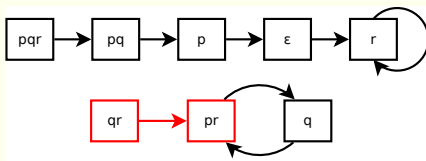


Learned Rules :
 $r \leftarrow \neg p \wedge \neg q \wedge r.$

Knowledge Base :
 $p \leftarrow p \wedge q.$
 $q \leftarrow p \wedge q \wedge r.$
 $r \leftarrow \neg p \wedge \neg q.$

Running Example

Input State Transitions



Learned Rules :

$$p \leftarrow \neg p \wedge q \wedge r.$$

$$r \leftarrow \neg p \wedge q \wedge r.$$

Knowledge Base :

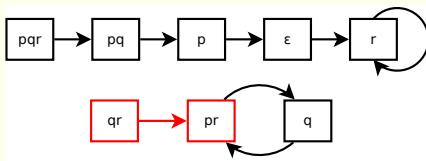
$$p \leftarrow p \wedge q.$$

$$q \leftarrow p \wedge q \wedge r.$$

$$r \leftarrow \neg p \wedge \neg q.$$

Running Example

Input State Transitions



Learned Rules :

$$p \leftarrow \neg p \wedge q \wedge r.$$

$$r \leftarrow \neg p \wedge q \wedge r.$$

Knowledge Base :

$$p \leftarrow p \wedge q.$$

$$q \leftarrow p \wedge q \wedge r.$$

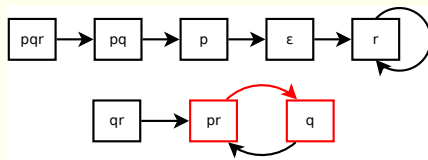
$$r \leftarrow \neg p \wedge \neg q.$$

$$p \leftarrow q \wedge r.$$

$$r \leftarrow \neg p \wedge r.$$

Running Example

Input State Transitions

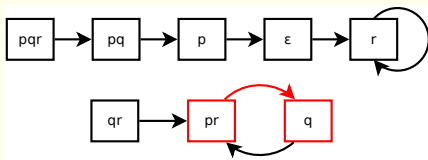


Learned Rules :
 $q \leftarrow p \wedge \neg q \wedge r.$

Knowledge Base :
 $p \leftarrow p \wedge q.$
 $q \leftarrow p \wedge q \wedge r.$
 $r \leftarrow \neg p \wedge \neg q.$
 $p \leftarrow q \wedge r.$
 $r \leftarrow \neg p \wedge r.$

Running Example

Input State Transitions

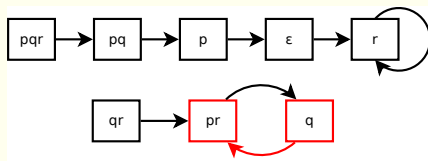


Learned Rules :
 $q \leftarrow p \wedge \neg q \wedge r.$

Knowledge Base :
 $p \leftarrow p \wedge q.$
 $q \leftarrow p \wedge r.$
 $r \leftarrow \neg p \wedge \neg q.$
 $p \leftarrow q \wedge r.$
 $r \leftarrow \neg p \wedge r.$

Running Example

Input State Transitions



Learned Rules :

$$p \leftarrow \neg p \wedge q \wedge \neg r.$$

$$r \leftarrow \neg p \wedge q \wedge \neg r.$$

Knowledge Base :

$$p \leftarrow p \wedge q.$$

$$q \leftarrow p \wedge r.$$

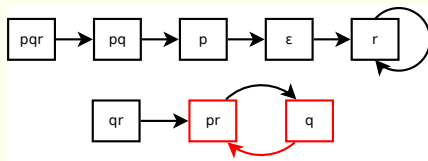
$$r \leftarrow \neg p \wedge \neg q.$$

$$p \leftarrow q \wedge r.$$

$$r \leftarrow \neg p \wedge r.$$

Running Example

Input State Transitions



Learned Rules :

$$p \leftarrow q \wedge \neg r.$$

$$r \leftarrow \neg p \wedge \neg r.$$

Knowledge Base :

$$p \leftarrow p \wedge q.$$

$$q \leftarrow p \wedge r.$$

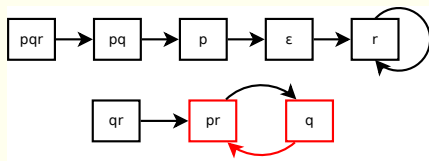
$$r \leftarrow \neg p \wedge \neg q.$$

$$p \leftarrow q \wedge r.$$

$$r \leftarrow \neg p \wedge r.$$

Running Example

Input State Transitions



Learned Rules :

$$p \leftarrow q \wedge \neg r.$$

$$r \leftarrow \neg p \wedge \neg r.$$

Knowledge Base :

$$p \leftarrow p \wedge q.$$

$$q \leftarrow p \wedge r.$$

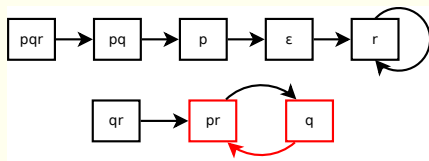
$$r \leftarrow \neg p \wedge \neg q.$$

$$p \leftarrow q.$$

$$r \leftarrow \neg p.$$

Running Example

Input State Transitions



Learned Rules :

$$p \leftarrow q \wedge \neg r.$$

$$r \leftarrow \neg p \wedge \neg r.$$

Knowledge Base :

$$q \leftarrow p \wedge r.$$

$$p \leftarrow q.$$

$$r \leftarrow \neg p.$$

LF1T with specialization

To guarantee to find minimal rules, LF1T starts with an initial program

$$P_0^B = \{p. | p \in B\}$$

LF1T learns a program P by analyzing each transition $(I, J) \in E$. For each variable A that does **not appear** in J , LF1T infers an **anti-rule**

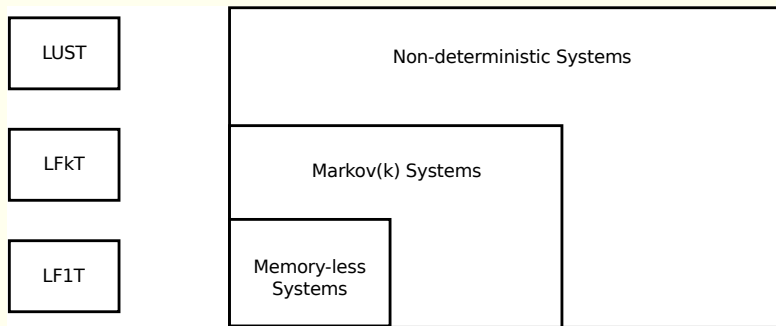
$$R_A^I := A \leftarrow \bigwedge_{B_i \in I} B_i \wedge \bigwedge_{C_j \in (B \setminus I)} \neg C_j$$

Exemple

From the state transition (bc, ac) we can learn 1 anti-rule : $b \leftarrow \neg a \wedge b \wedge c$.
If that rule is subsumed by P then P is not consistent with E .

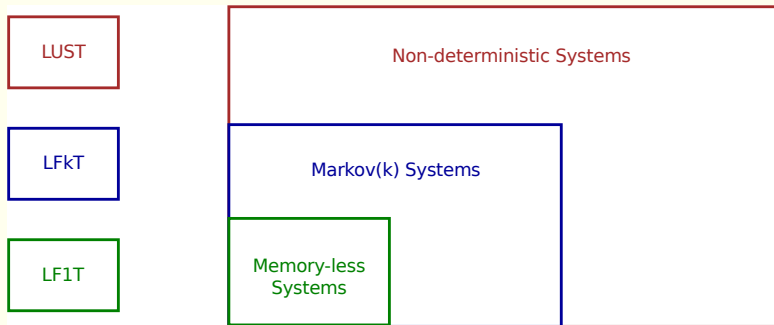
LFIT Algorithms

Three algorithms to capture three type of dynamics.



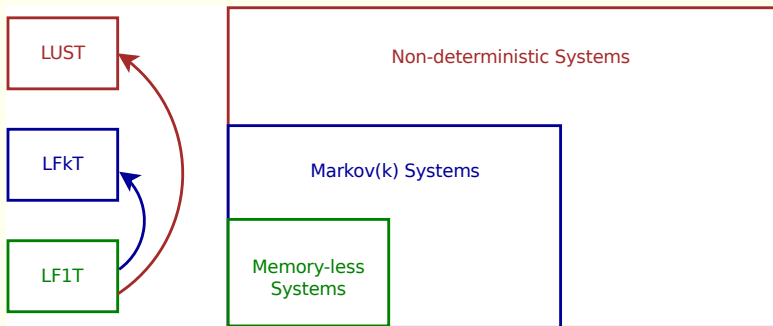
LFIT Algorithms

Three algorithms to capture three type of dynamics.

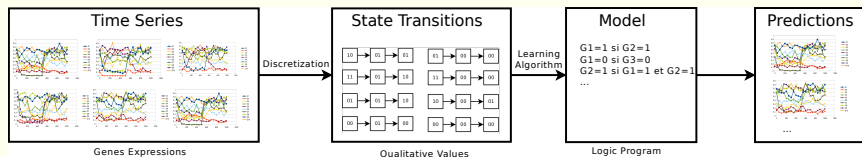


LFIT Algorithms

Three algorithms to capture three type of dynamics.



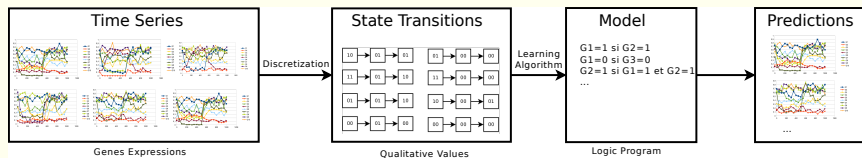
Learning Dynamics of Biological Systems



Process :

- Input : Time series of gene expressions

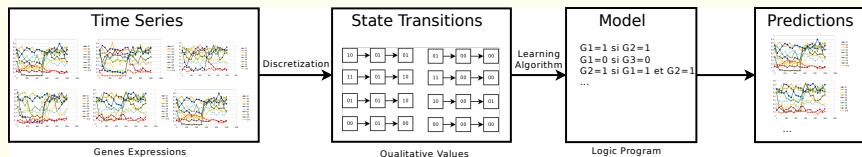
Learning Dynamics of Biological Systems



Process :

- Input : Time series of gene expressions
- Discretization : qualitative state transitions

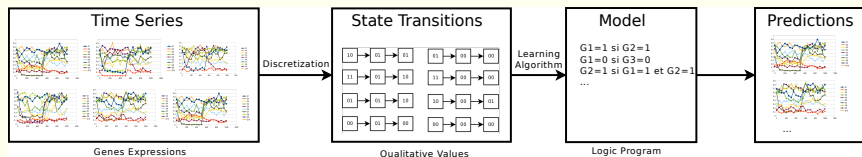
Learning Dynamics of Biological Systems



Process :

- Input : Time series of gene expressions
- Discretization : qualitative state transitions
- Learning : model of the dynamics as logic rules

Learning Dynamics of Biological Systems



Process :

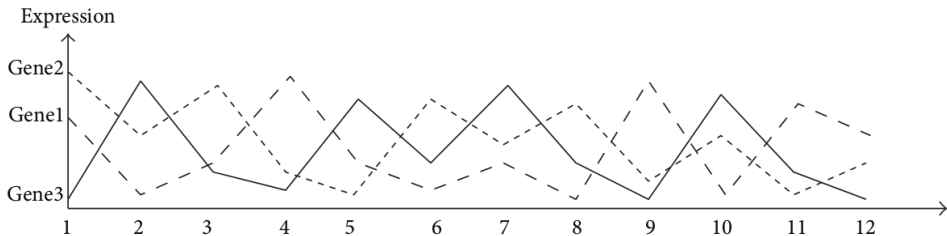
- Input : Time series of gene expressions
- Discretization : qualitative state transitions
- Learning : model of the dynamics as logic rules
- Usage : prediction of non-observed gene expressions

Sommaire

- 1 Introduction
- 2 Généralités — sur la représentation du temps dans les modèles de la dynamique des système
- 3 »»» PREMIERE PARTIE : Des modèles hybrides «««
- 4 Model-checking temporisé
- 5 Model-checking hybride
- 6 Abstraction et raffinement
- 7 »»» DEUXIEME PARTIE : Des données aux modèles «««
- 8 Apprentissage des modèles
- 9 Extension hybride : du discret ... aux intervalles**
- 10 Conclusion
- 11 Bibliographie

Abstraction : From Time Series Data to State Transitions

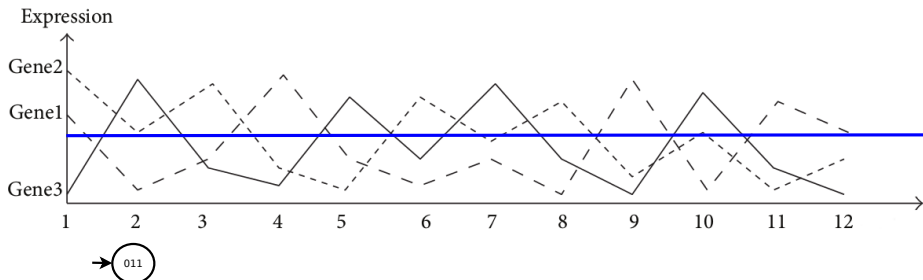
Time series data can be abstracted in many ways. It depends of what we know and what we want to learn about the system.



Time series data of gene expression (*Nakajima and Akutsu 2014*).

Abstraction : From Time Series Data to State Transitions

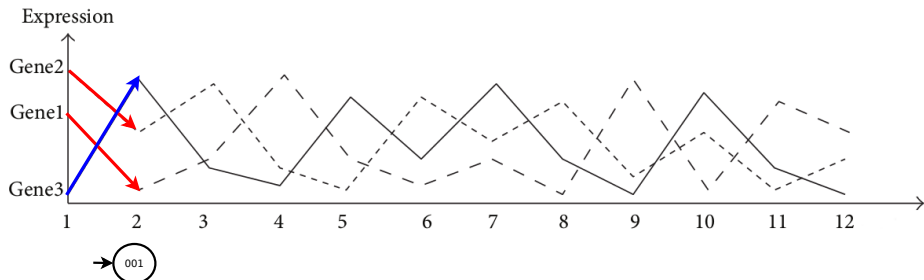
Time series data can be abstracted in many ways. It depends of what we know and what we want to learn about the system.



We can set thresholds to discretize the level of expression of genes.

Abstraction : From Time Series Data to State Transitions

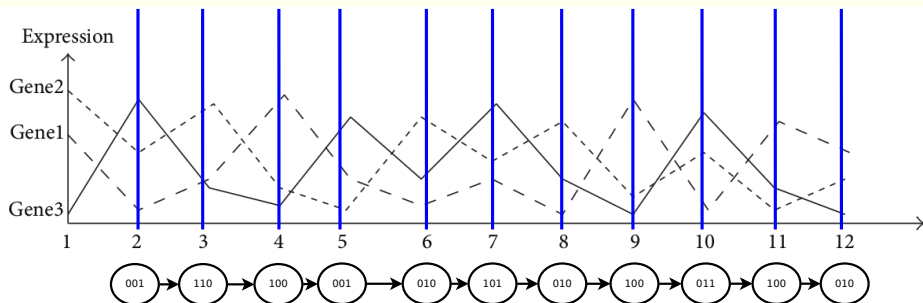
Time series data can be abstracted in many ways. It depends of what we know and what we want to learn about the system.



We can consider the concentration behavior (increase/decrease).

Abstraction : From Time Series Data to State Transitions

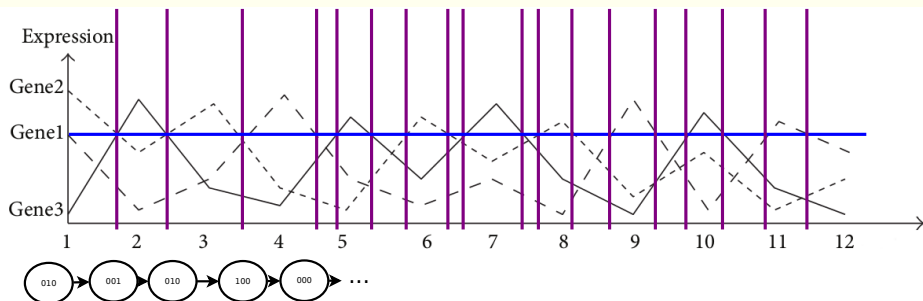
Time series data can be abstracted in many ways. It depends of what we know and what we want to learn about the system.



Transitions can be **time-based** : a state for each time unit.

Abstraction : From Time Series Data to State Transitions

Time series data can be abstracted in many ways. It depends of what we know and what we want to learn about the system.



Transitions can be **event-based** : a state for each change.

LUST/ACEDIA interval

INPUT generation

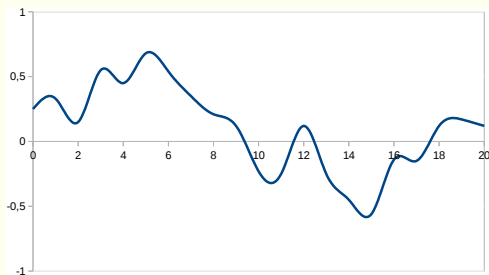
- A dynamic system
- Generate curves ($\in [-1, 1]$)
- Sampling

- INPUT : a set of time series S ,
 $d = 2, P = \emptyset$
- 1) Discretize S according to d ,
into a set of traces E
- 2) Complete P with
 $LUST/ACEDIA(P, E)$
- 3) Double d , go to 1)

LUST/ACEDIA interval

INPUT generation

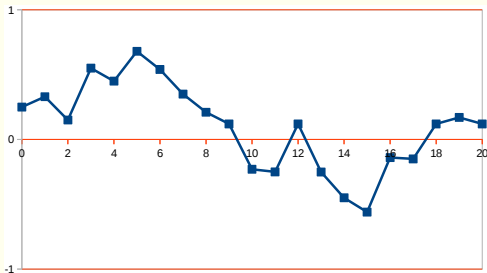
- A dynamic system
- Generate curves ($\in [-1, 1]$)
- Sampling
- INPUT : a set of time series S ,
 $d = 2$, $P = \emptyset$
- 1) Discretize S according to d ,
into a set of traces E
- 2) Complete P with
 $LUST/ACEDIA(P, E)$
- 3) Double d , go to 1)



LUST/ACEDIA interval

INPUT generation

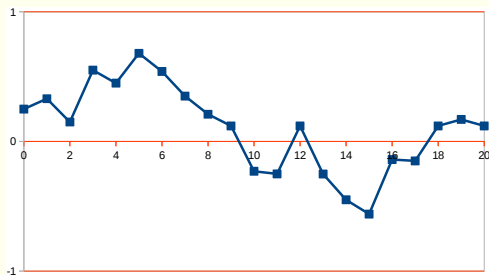
- A dynamic system
- Generate curves ($\in [-1, 1]$)
- Sampling
- INPUT : a set of time series S ,
 $d = 2$, $P = \emptyset$
- 1) Discretize S according to d ,
into a set of traces E
- 2) Complete P with
 $LUST/ACEDIA(P, E)$
- 3) Double d , go to 1)



LUST/ACEDIA interval

INPUT generation

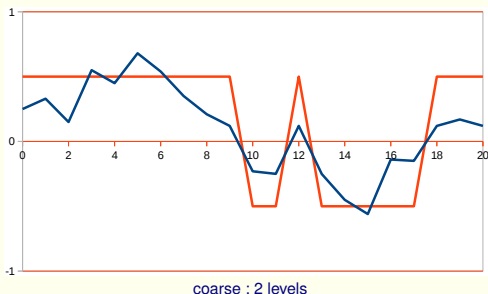
- A dynamic system
- Generate curves ($\in [-1, 1]$)
- Sampling
- INPUT : a set of time series S ,
 $d = 2$, $P = \emptyset$
- 1) Discretize S according to d ,
into a set of traces E
- 2) Complete P with
 $LUST/ACEDIA(P, E)$
- 3) Double d , go to 1)



LUST/ACEDIA interval

INPUT generation

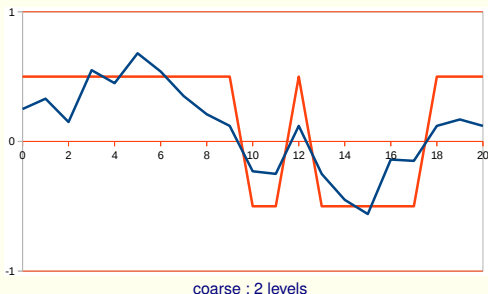
- A dynamic system
- Generate curves ($\in [-1, 1]$)
- Sampling
- INPUT : a set of time series S ,
 $d = 2$, $P = \emptyset$
- 1) Discretize S according to d ,
into a set of traces E
- 2) Complete P with
 $LUST/ACEDIA(P, E)$
- 3) Double d , go to 1)



LUST/ACEDIA interval

INPUT generation

- A dynamic system
- Generate curves ($\in [-1, 1]$)
- Sampling
- INPUT : a set of time series S ,
 $d = 2$, $P = \emptyset$
- 1) Discretize S according to d ,
into a set of traces E
- 2) Complete P with
 $LUST/ACEDIA(P, E)$
- 3) Double d , go to 1)

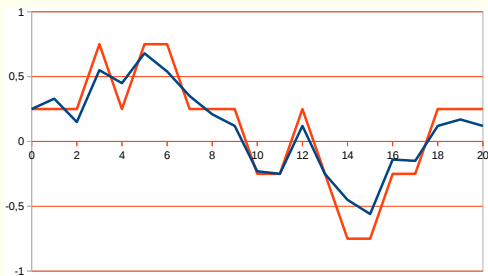


$a([0, 1], T) :: 10/12 :- a([0, 1], T - 1)$
 $a([0, 1], T) :: 2/7 :- a([-1, 0], T - 1)$
 $a([-1, 0], T) :: 2/12 :- a([0, 1], T - 1)$
 $a([-1, 0], T) :: 5/7 :- a([-1, 0], T - 1)$
 ...

LUST/ACEDIA interval

INPUT generation

- A dynamic system
- Generate curves ($\in [-1, 1]$)
- Sampling
- INPUT : a set of time series S ,
 $d = 2$, $P = \emptyset$
- 1) Discretize S according to d ,
into a set of traces E
- 2) Complete P with
 $LUST/ACEDIA(P, E)$
- 3) Double d , go to 1)



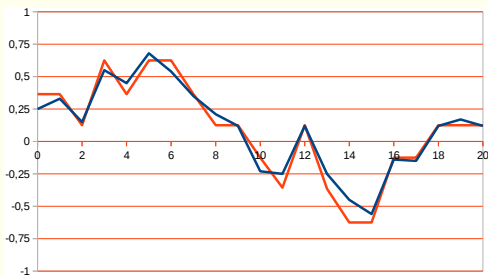
refined : 4 levels

$a([0.5, 1], T) : :1/3 :- a([0.5, 1], T - 1)$
 $a([0.5, 1], T) : :2/11 :- a([0, 0.5], T - 1)$
 $a([0, 0.5], T) : :2/3 :- a([0.5, 1], T - 1)$
 $a([0, 0.5], T) : :6/11 :- a([0, 0.5], T - 1)$
 $a([0, 0.5], T) : :2/4 :- a([-0.5, 0], T - 1)$
 ...

LUST/ACEDIA interval

INPUT generation

- A dynamic system
- Generate curves ($\in [-1, 1]$)
- Sampling
- INPUT : a set of time series S ,
 $d = 2$, $P = \emptyset$
- 1) Discretize S according to d ,
into a set of traces E
- 2) Complete P with
 $LUST/ACEDIA(P, E)$
- 3) Double d , go to 1)



refined : 8 levels

$$\begin{aligned}
 &a([0.5, 0.75], T) : :1/3 :- a([0.5, 0.75], T - 1) \\
 &a([0.5, 0.75], T) : :1/3 :- a([0.25, 0.5], T - 1) \\
 &a([0.25, 0.5], T) : :1/3 :- a([0.5, 0.75], T - 1) \\
 &a([0.25, 0.5], T) : :1/3 :- a([0.25, 0.5], T - 1) \\
 &a([0, 0.25], T) : :1/3 :- a([0.5, 0.75], T - 1) \\
 &a([0, 0.25], T) : :1/3 :- a([0.5, 0.75], T - 1) \\
 &a([0, 0.25], T) : :3/6 :- a([0.25, 0.5], T - 1) \\
 &a([0, 0.25], T) : :1/3 :- a([-0.25, 0], T - 1)
 \end{aligned}$$

...

LUST/ACEDIA interval

INPUT generation

- A dynamic system
- Generate curves ($\in [-1, 1]$)
- Sampling
- INPUT : a set of time series S ,
 $d = 2$, $P = \emptyset$
- 1) Discretize S according to d ,
into a set of traces E
- 2) Complete P with
 $LUST/ACEDIA(P, E)$
- 3) Double d , go to 1)

$$\begin{aligned}
 a([0, 1], T) &:: 10/12 :- a([0, 1], T - 1) \\
 a([0, 1], T) &:: 2/7 :- a([-1, 0], T - 1) \\
 a([-1, 0], T) &:: 2/12 :- a([0, 1], T - 1) \\
 a([-1, 0], T) &:: 5/7 :- a([-1, 0], T - 1) \\
 &\dots
 \end{aligned}$$

$$\begin{aligned}
 a([0.5, 1], T) &:: 1/3 :- a([0.5, 1], T - 1) \\
 a([0.5, 1], T) &:: 2/11 :- a([0, 0.5], T - 1) \\
 a([0, 0.5], T) &:: 2/3 :- a([0.5, 1], T - 1) \\
 a([0, 0.5], T) &:: 6/11 :- a([0, 0.5], T - 1) \\
 a([0, 0.5], T) &:: 2/4 :- a([-0.5, 0], T - 1) \\
 &\dots
 \end{aligned}$$

$$\begin{aligned}
 a([0.5, 0.75], T) &:: 1/3 :- a([0.5, 0.75], T - 1) \\
 a([0.5, 0.75], T) &:: 1/3 :- a([0.25, 0.5], T - 1) \\
 a([0.25, 0.5], T) &:: 1/3 :- a([0.5, 0.75], T - 1) \\
 a([0.25, 0.5], T) &:: 1/3 :- a([0.25, 0.5], T - 1) \\
 a([0, 0.25], T) &:: 1/3 :- a([0.5, 0.75], T - 1) \\
 a([0, 0.25], T) &:: 1/3 :- a([0.5, 0.75], T - 1) \\
 a([0, 0.25], T) &:: 3/6 :- a([0.25, 0.5], T - 1) \\
 a([0, 0.25], T) &:: 1/3 :- a([-0.25, 0], T - 1) \\
 &\dots
 \end{aligned}$$

LUST/ACEDIA interval postprocessing

Rules can be combined on both conclusion and conditions.

Exemple (Head fusion)

$$\begin{aligned} a([0.25, 0.5], T) &: :1/3 :- b([0.5, 0.75], T - 1) \\ a([0.5, 0.75], T) &: :2/3 :- b([0.5, 0.75], T - 1) \end{aligned}$$

Exemple (Body fusion)

$$\begin{aligned} a([0.25, 0.75], T) &: :1/3 :- b([0.5, 0.75], T - 1) \\ a([0.25, 0.75], T) &: :2/3 :- b([0.25, 0.5], T - 1) \end{aligned}$$

LUST/ACEDIA interval postprocessing

Rules can be combined on both conclusion and conditions.

Exemple (Head fusion)

$$a([0.25, 0.5], T) : :1/3 :- b([0.5, 0.75], T - 1)$$

$$a([0.5, 0.75], T) : :2/3 :- b([0.5, 0.75], T - 1)$$

$$a([0.25, 0.75], T) : :3/3 :- b([0.5, 0.75], T - 1)$$

Exemple (Body fusion)

$$a([0.25, 0.75], T) : :1/3 :- b([0.5, 0.75], T - 1)$$

$$a([0.25, 0.75], T) : :2/3 :- b([0.25, 0.5], T - 1)$$

LUST/ACEDIA interval postprocessing

Rules can be combined on both conclusion and conditions.

Exemple (Head fusion)

$$a([0.25, 0.5], T) : : 1/3 :- b([0.5, 0.75], T - 1)$$

$$a([0.5, 0.75], T) : : 2/3 :- b([0.5, 0.75], T - 1)$$

$$a([0.25, 0.75], T) : : 3/3 :- b([0.5, 0.75], T - 1)$$

Exemple (Body fusion)

$$a([0.25, 0.75], T) : : 1/3 :- b([0.5, 0.75], T - 1)$$

$$a([0.25, 0.75], T) : : 2/3 :- b([0.25, 0.5], T - 1)$$

LUST/ACEDIA interval postprocessing

Rules can be combined on both conclusion and conditions.

Exemple (Head fusion)

$$a([0.25, 0.5], T) : :1/3 :- b([0.5, 0.75], T - 1)$$

$$a([0.5, 0.75], T) : :2/3 :- b([0.5, 0.75], T - 1)$$

$$a([0.25, 0.75], T) : :3/3 :- b([0.5, 0.75], T - 1)$$

Exemple (Body fusion)

$$a([0.25, 0.75], T) : :1/3 :- b([0.5, 0.75], T - 1)$$

$$a([0.25, 0.75], T) : :2/3 :- b([0.25, 0.5], T - 1)$$

LUST/ACEDIA interval postprocessing

Rules can be combined on both conclusion and conditions.

Exemple (Head fusion)

$$a([0.25, 0.5], T) : : 1/3 :- b([0.5, 0.75], T - 1)$$

$$a([0.5, 0.75], T) : : 2/3 :- b([0.5, 0.75], T - 1)$$

$$a([0.25, 0.75], T) : : 3/3 :- b([0.5, 0.75], T - 1)$$

Exemple (Body fusion)

$$a([0.25, 0.75], T) : : 1/3 :- b([0.5, 0.75], T - 1)$$

$$a([0.25, 0.75], T) : : 2/3 :- b([0.25, 0.5], T - 1)$$

LUST/ACEDIA interval postprocessing

Rules can be combined on both conclusion and conditions.

Exemple (Head fusion)

$$a([0.25, 0.5], T) : :1/3 :- b([0.5, 0.75], T - 1)$$

$$a([0.5, 0.75], T) : :2/3 :- b([0.5, 0.75], T - 1)$$

$$a([0.25, 0.75], T) : :3/3 :- b([0.5, 0.75], T - 1)$$

Exemple (Body fusion)

$$a([0.25, 0.75], T) : :1/3 :- b([0.5, 0.75], T - 1)$$

$$a([0.25, 0.75], T) : :2/3 :- b([0.25, 0.5], T - 1)$$

$$a([0.25, 0.75], T) : :3/3 :- b([0.25, 0.75], T - 1)$$

LUST/ACEDIA interval postprocessing

Rules can be combined on both conclusion and conditions.

Exemple (Head fusion)

$$a([0.25, 0.5], T) : :1/3 :- b([0.5, 0.75], T - 1)$$

$$a([0.5, 0.75], T) : :2/3 :- b([0.5, 0.75], T - 1)$$

$$a([0.25, 0.75], T) : :3/3 :- b([0.5, 0.75], T - 1)$$

Exemple (Body fusion)

$$a([0.25, 0.75], T) : :1/3 :- b([0.5, 0.75], T - 1)$$

$$a([0.25, 0.75], T) : :2/3 :- b([0.25, 0.5], T - 1)$$

$$a([0.25, 0.75], T) : :3/3 :- b([0.25, 0.75], T - 1)$$

Interval fusion outcomes

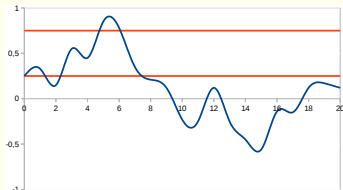
Interval fusions give **expression levels** !

Exemple (Body fusion)

$a([0.25, 0.75], T) : :1/3 :- b([0.5, 0.75], T - 1)$

$a([0.25, 0.75], T) : :2/3 :- b([0.25, 0.5], T - 1)$

$a([0.25, 0.75], T) : :3/3 :- b([0.25, 0.75], T - 1)$



Sommaire

- 1 Introduction
- 2 Généralités — sur la représentation du temps dans les modèles de la dynamique des système
- 3 »»» PREMIERE PARTIE : Des modèles hybrides «««
- 4 Model-checking temporisé
- 5 Model-checking hybride
- 6 Abstraction et raffinement
- 7 »»» DEUXIEME PARTIE : Des données aux modèles «««
- 8 Apprentissage des modèles
- 9 Extension hybride : du discret ... aux intervalles
- 10 Conclusion**
- 11 Bibliographie

Sur notre démarche hybride

Nous avons vu :

- ▶ l'introduction du temps dans la modélisation et le model-checking temporisé et hybride
- ▶ l'apprentissage à partir de données de séries temporelles

Sur notre démarche hybride

Nous avons vu :

- ▶ l'introduction du temps dans la modélisation et le model-checking temporisé et hybride
- ▶ l'apprentissage à partir de données de séries temporelles

Ce qu'on peut retenir :

- ▶ de l'importance de l'abstraction et du raffinement
- ▶ la fusion des intervalles donne les niveaux d'expression
- ▶ le temps est une donnée comme les autres... et les intervalles de temps donneront les délais

Sommaire

- 1 Introduction
- 2 Généralités — sur la représentation du temps dans les modèles de la dynamique des système
- 3 »»» PREMIERE PARTIE : Des modèles hybrides «««
- 4 Model-checking temporisé
- 5 Model-checking hybride
- 6 Abstraction et raffinement
- 7 »»» DEUXIEME PARTIE : Des données aux modèles «««
- 8 Apprentissage des modèles
- 9 Extension hybride : du discret ... aux intervalles
- 10 Conclusion
- 11 Bibliographie



J. Ahmad, G. Bernot, J.-P. Comet, D. Lime, and O. Roux.
Hybrid modelling and dynamical analysis of gene regulatory networks with delays.
ComPlexUs, 3(4) :231–251, 2006 (Cover Date : November 2007).



Anil Aswani and Claire Tomlin.
Reachability algorithm for biological piecewise-affine hybrid systems.
In *HSCC'07 : Proceedings of the 10th international conference on Hybrid systems*, pages 633–636, Berlin, Heidelberg, 2007. Springer-Verlag.



Grégory Batt, Calin Belta, and Ron Weiss.
Model checking genetic regulatory networks with parameter uncertainty.
In *HSCC'07 : Proceedings of the 10th international conference on Hybrid systems*, pages 61–75, Berlin, Heidelberg, 2007. Springer-Verlag.



Grégory Batt, Calin Belta, and Ron Weiss.
Model checking liveness properties of genetic regulatory networks.
In *TACAS'07 : Proceedings of the 13th international conference on Tools and algorithms for the construction and analysis of systems*, pages 323–338, Berlin, Heidelberg, 2007. Springer-Verlag.



G. Bernot, J.-P. Comet, A. Richard, and J. Guespin.
Application of formal methods to biological regulatory networks : Extending Thomas' asynchronous logical approach with temporal logic.
Journal of Theoretical Biology, 229(3) :339–347, 2004.



L. Bortolussi and A. Policriti.
Hybrid systems and biology. continuous and discrete modeling for systems biology.
In P. Degano M. Bernardo and G. Zavattaro, editors, *Formal Methods For Computational*