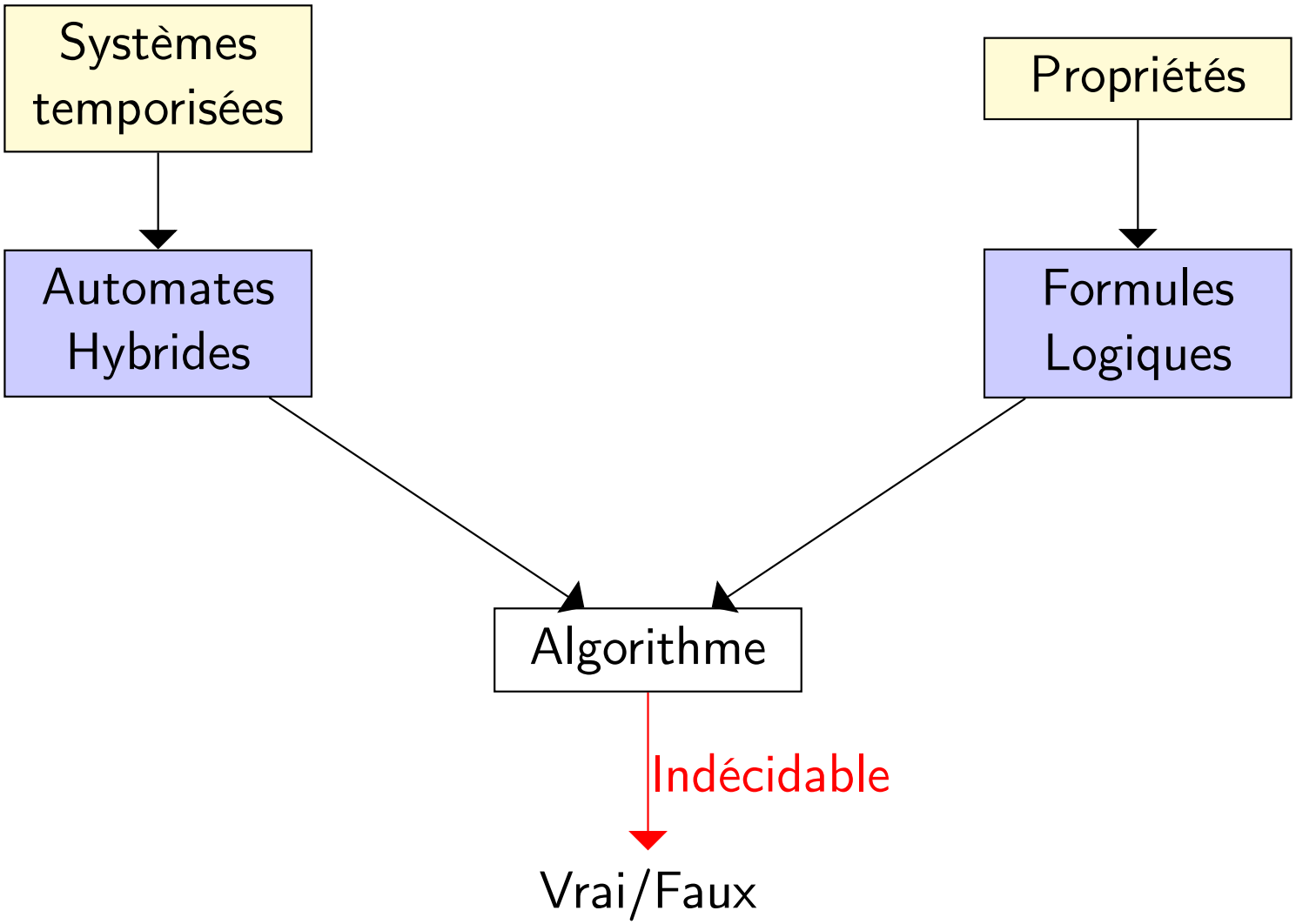
Application de méthodes d'algèbre effective pour l'étude d'automates temporisés

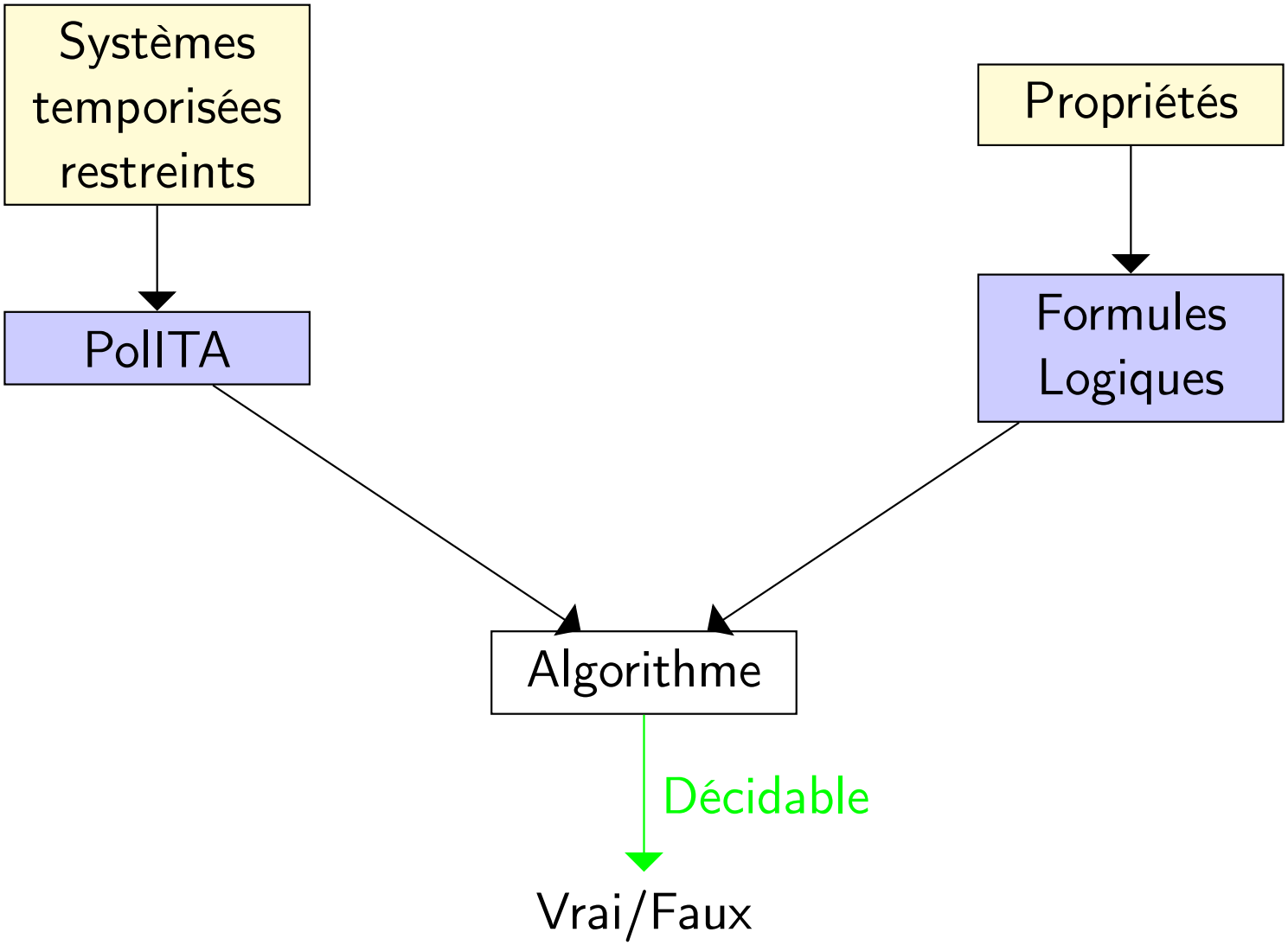
Rémy Garnier et Mathieu Huot

ENS Cachan

30 juin 2015

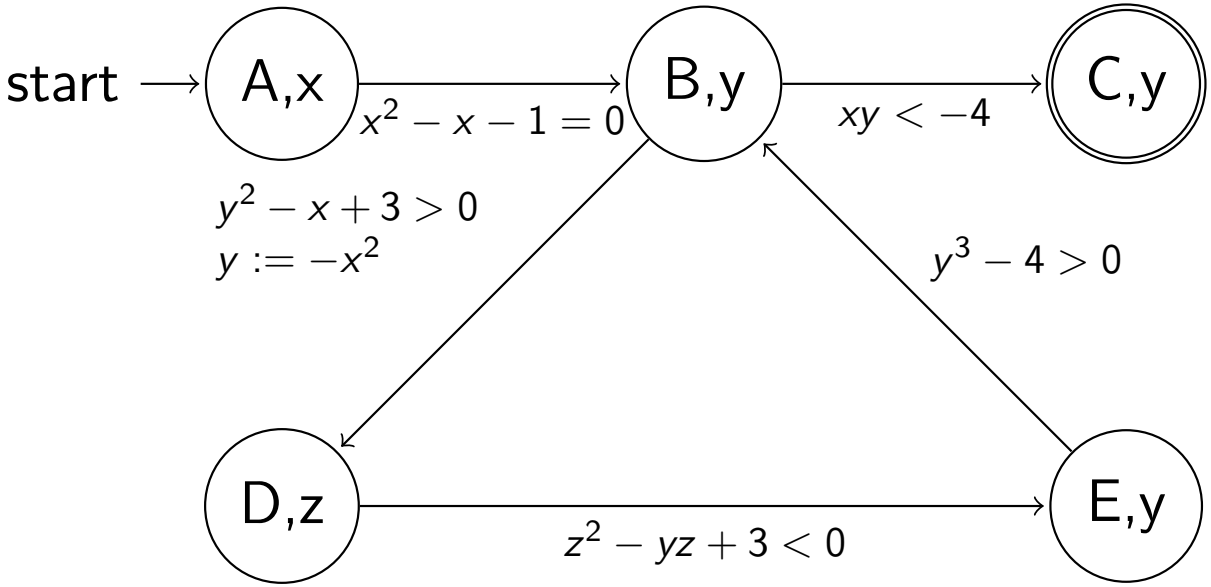
Plan


Systèmes temporisées
Propriétés
Automates Hybrides
Formules Logiques
Algorithme
Indécidable
Vrai/Faux


Systèmes temporisées restreints
Propriétés
PolITA
Formules Logiques
Algorithme
Décidable
Vrai/Faux

PolITA

- Ensemble *ordonné* d'horloges $X = \{x, y, z, ...\}$: paramètres du système
- Ensemble fini Q d'états : configurations du système
- Transitions entre ces états sous des conditions polynomiales



Le problème de l'accessibilité

Dans le système de contrôle : Peut-on atteindre tel état ?

- Système de transitions infini $Q \times \mathbb{R}^X$:
  accessibilité en temps fini ?
  $\Rightarrow$ Produit synchronisé $\{(q, C) | q \in Q, C \text{ région de } \mathbb{R}^X\}$

Le problème de l'accessibilité

Dans le système de contrôle : Peut-on atteindre tel état ?

- Système de transitions infini $Q \times \mathbb{R}^X$:
  accessibilité en temps fini ?
  $\Rightarrow$ Produit synchronisé $\{(q, C) | q \in Q, C \text{ région de } \mathbb{R}^X\}$
- Problème : partition de l'espace en un nombre fini de régions de $\mathbb{R}^X$
  $\Rightarrow$ Calculé par la décomposition cylindrique

Décomposition Cylindrique

$\mathcal{P} \subset \mathbb{Q}[X_1, X_2, \ldots, X_n]$

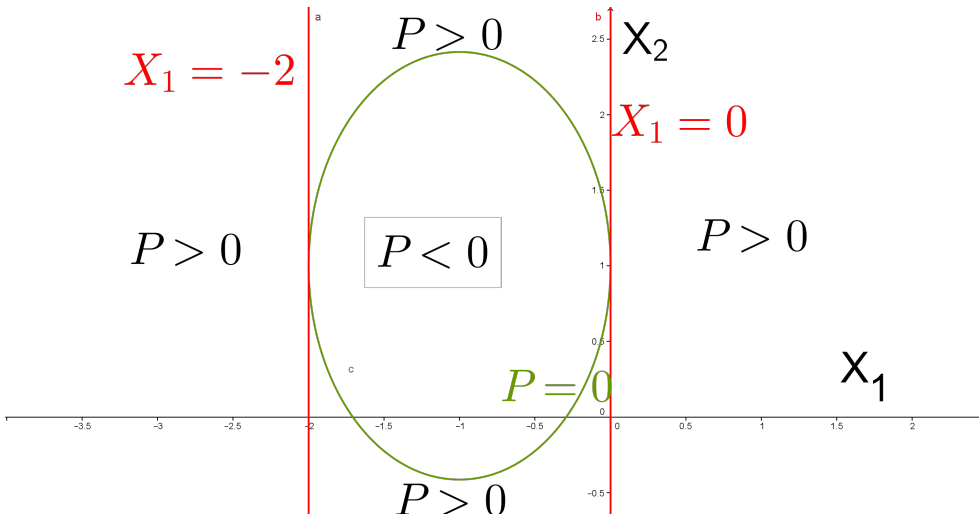
Principe : découpage de l'espace $\mathbb{R}^n$ selon des zones dans lesquelles les polynômes de $\mathcal{P}$ gardent un signe constant.

Exemple très simple (ellipse)

$P = (X_1 + 1)^2 + \frac{(X_2 - 1)^2}{2} - 1 < 0$

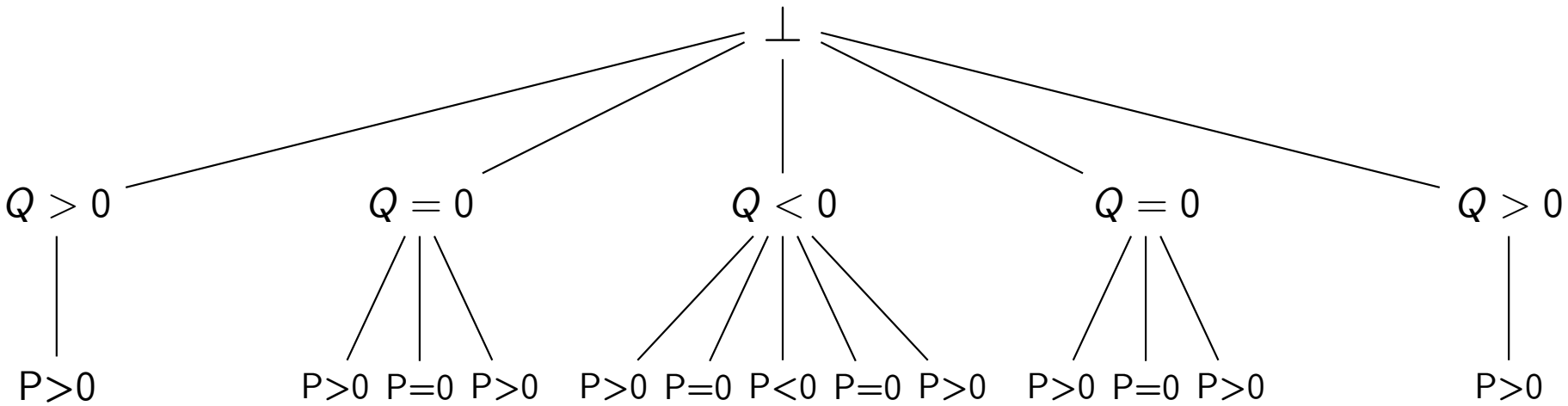
L'algorithme génère le polynôme :

$Q = sRes(P, P') = X_1^2 + 2 \cdot X_1$



Arbre de Décomposition cylindrique

$$P = (X_1 + 1)^2 + \frac{(X_2 - 1)^2}{2} - 1 < 0 \quad Q = sRes(P, P') = X_1^2 + 2 \cdot X_1$$



Améliorations des algorithmes

Complexité des algorithmes très élevée (2EXPTIME)
$\Rightarrow$ Nécessité d'améliorations des algorithmes.

Améliorations des algorithmes

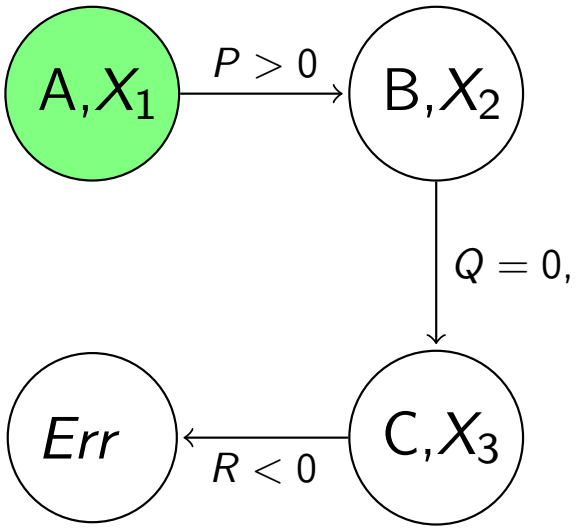
Complexité des algorithmes très élevée (2EXPTIME)
$\Rightarrow$ Nécessité d'améliorations des algorithmes. Parmi elles :

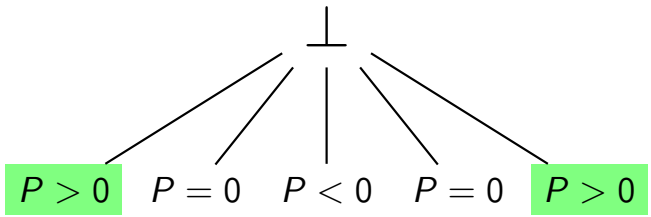
- Parallélisation du code
- Différents pré-calculs
- Choix de la division exacte
- Simplification des polynômes après phase d'élimination
- Calcul du produit synchronisé "à-la-volée"

Produit synchronisé à-la-volée

Principe : ne construire que les parties de l'arbre qui nous intéressent

$\Rightarrow$ Évite d'avoir à calculer tout l'arbre.

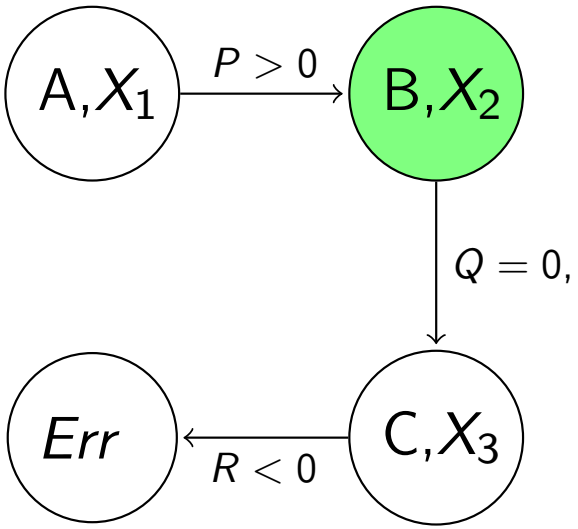


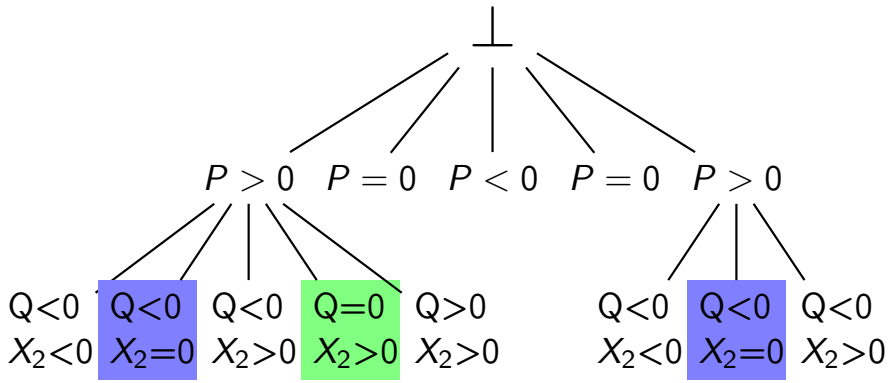


Produit synchronisé à-la-volée

Principe : ne construire que les parties de l'arbre qui nous intéressent

$\Rightarrow$ Évite d'avoir à calculer tout l'arbre.

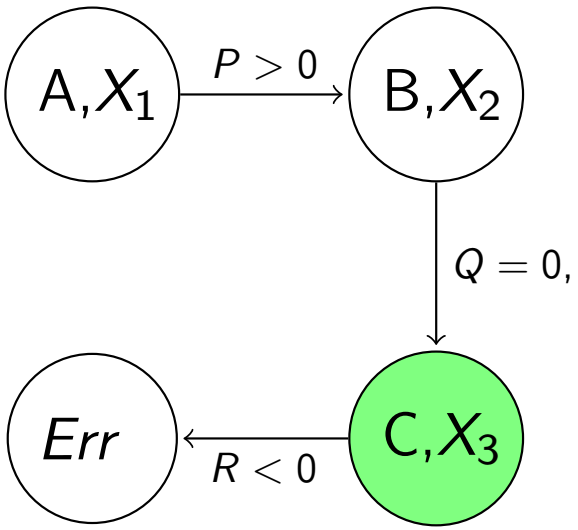


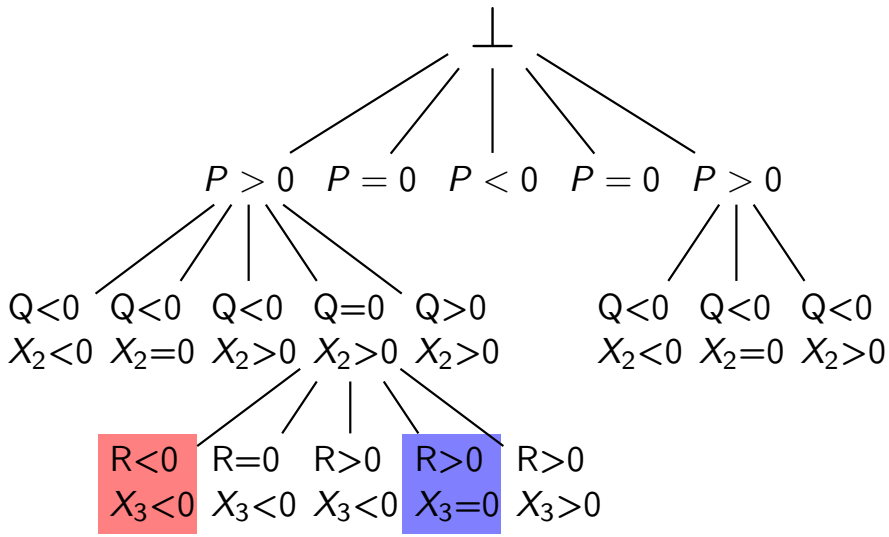


Produit synchronisé à-la-volée

Principe : ne construire que les parties de l'arbre qui nous intéressent

$\Rightarrow$ Évite d'avoir à calculer tout l'arbre.





Conclusion et perspectives

L'algorithme :

- implémenté
- tests
- améliorations

Conclusion et perspectives

L'algorithme :

- implémenté
- tests
- améliorations

Intégration dans l'environnement Cosy-Vérif :

- travail en cours avec A. Linard, ingénieur INRIA et A. Martin, stagiaire UPMC
- interface Python-Lua

Conclusion et perspectives

L'algorithme :

- implémenté
- tests
- améliorations

Intégration dans l'environnement Cosy-Vérif :

- travail en cours avec A. Linard, ingénieur INRIA et A. Martin, stagiaire UPMC
- interface Python-Lua

Perspectives : cas d'étude réels

Merci de votre attention