Vérification distribuée à la volée de grands espaces d'états

Christophe Joubert

INRIA Rhône-Alpes / VASY
http://www.inrialpes.fr/vasy

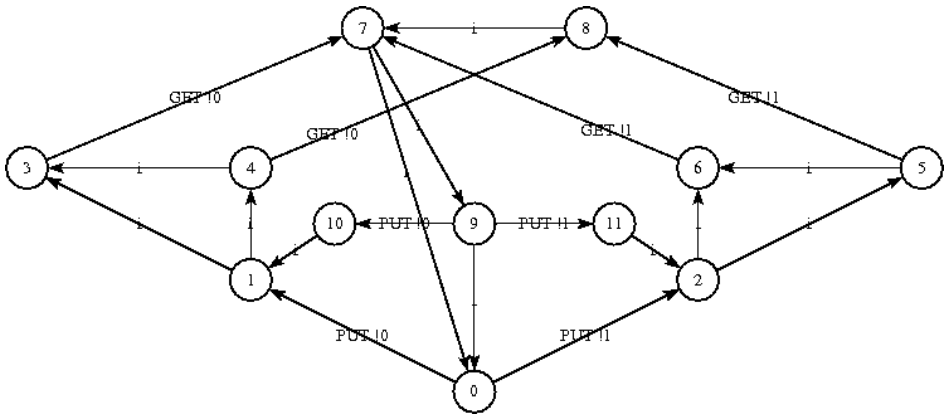
12 décembre 2005
Thèse réalisée sous la direction de
Radu Mateescu et Hubert Garavel

Vérification formelle

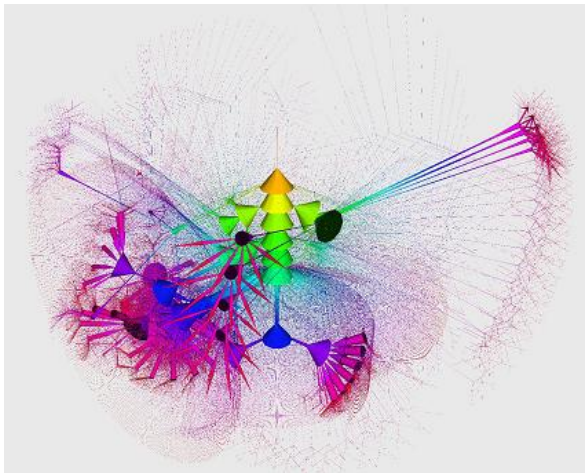
- Objectif : produire des logiciels fiables
- Mise en œuvre : utiliser des modèles formels et les capacités de calcul des ordinateurs pour analyser leur comportement
- Cibles : systèmes informatiques critiques, impliquant coût humain ou financier élevé
- Exemple : perte du satellite Cryosat - 08/10/05 - erreur logicielle sur le lanceur Rockot - 136 M €

Modèle formel : système de transitions étiquetées (STE)

- Comportement simplifié d'un protocole d'échange de données entre deux ordinateurs :

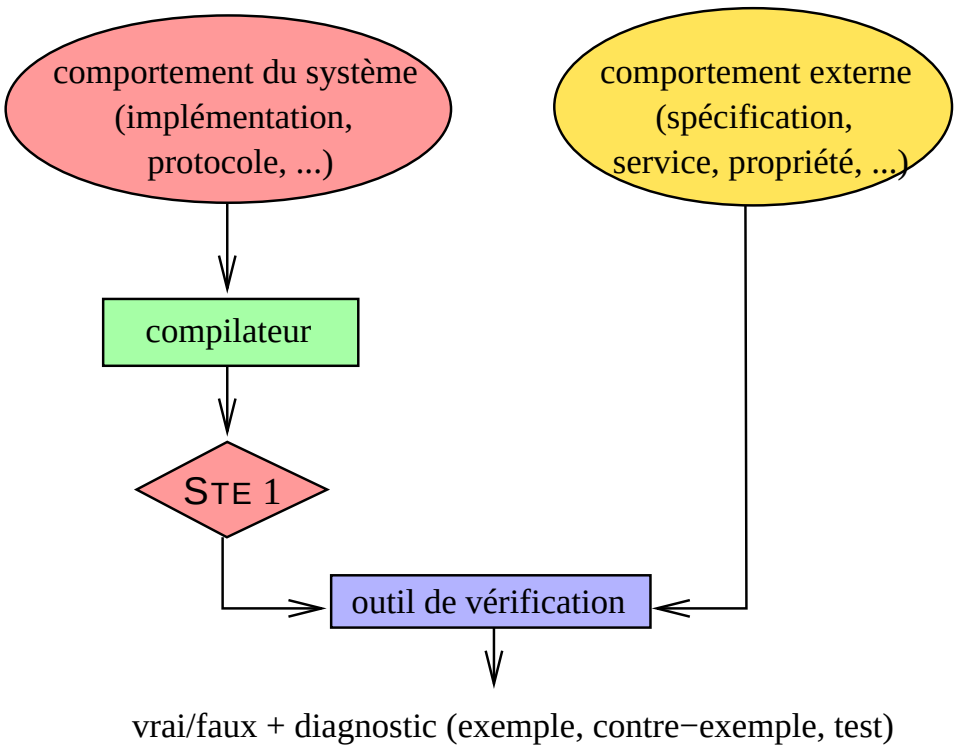


- STE de taille réaliste (10^5 états, 10^5 transitions) issue de la base de tests VLTS :



- Support logiciel (CADP) pour la représentation de STE :
  - *explicite* (fonction prédécesseur/successeur) – BCG (Binary Coded Graph)
  - *implicite* (fonction successeur) – OPEN/CÆSAR [Garavel-98]

Vérification énumérative

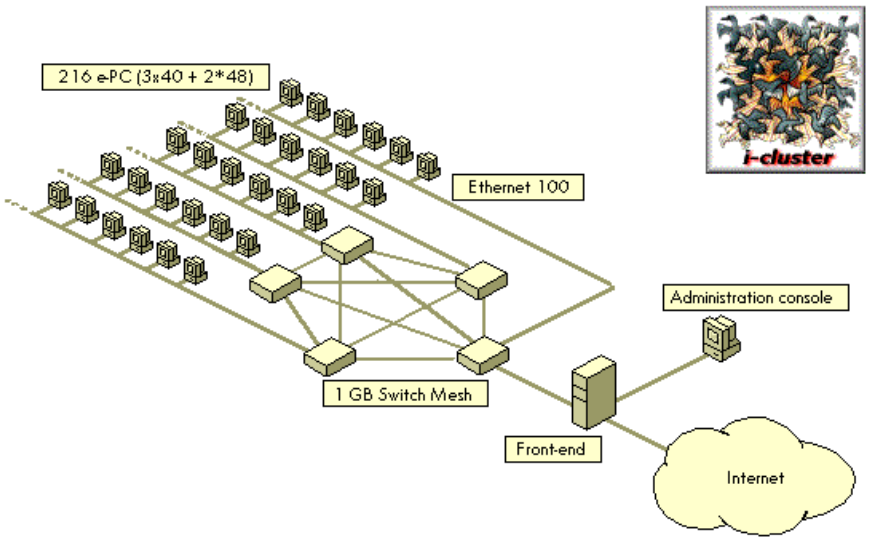


- **Vérification globale**
  - STE construit avant la vérification
- **Vérification à la volée**
  - STE construit pendant la vérification
  - Possibilité d'exploration partielle du STE pour obtenir le résultat
- **Problème d'explosion d'états**

Vérification distribuée

- Utiliser la puissance de calcul et l'espace mémoire de machines interconnectées pour résoudre des problèmes complexes

- ICLUSTER (INRIA/ID)
  216 PIII 733 MHz 256 Mb



- IDPOT
  48 Bi-Xeon 2.5 GHz 1.5 Gb

http://www.grid5000.org/

Grid'5000

Quatre grands problèmes abordés dans la thèse

Vérification énumérative

- Comparaison à la volée par équivalence
- Minimisation à la volée (τ-confluence)
- Evaluation à la volée de formules de logique temporelle

Génération de test

- Génération à la volée de cas de tests de conformité

Approche générique aux quatre grands problèmes

$\Rightarrow$ Résolution de systèmes d'équations booléennes (SEBS) avec diagnostic

Vérification énumérative

- Relations d'équivalences
  `[Andersen-Vergauwen-95],[Mateescu-03]`
- τ-confluence `[Pace-Lang-Mateescu-03]`
- Formules de μ-calcul modal
  `[Andersen-94],[Mateescu-Sighireanu-02]`

Génération de test

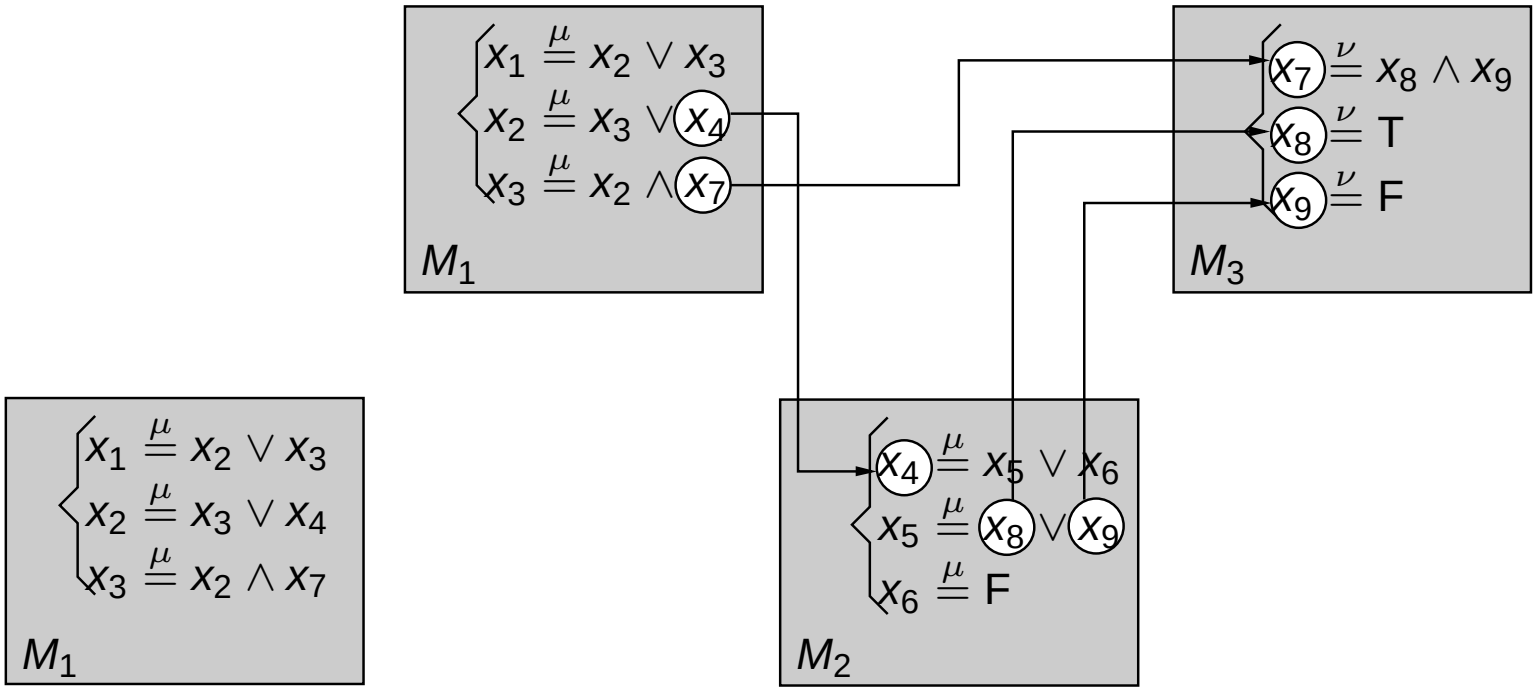
- Cas de tests conformes

Plan de l'exposé

1. Systèmes d'Equations Booléennes
2. Résolution distribuée à la volée de SEBs
3. Trois applications en vérification énumérative
4. Application à la génération de tests
5. Conclusion et perspectives

Plan de l’exposé

SEB monobloc et multiblocs



- Ensemble d'équations booléennes de point fixe ($M_i = \{x_j \overset{\sigma_i}{=} op_j X_j\}_{1 \leq j \leq m_i, 1 \leq i \leq n}$)
- Formules purement disjonctives ou conjonctives (SEB simple)
- n blocs M_i ($i \in [1..n]$) avec dépendances inter-blocs acycliques

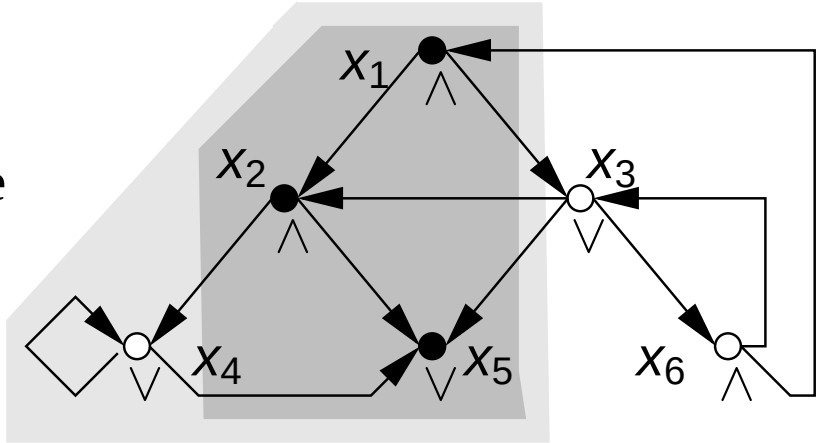
Graphe booléen et résolution de SEB

○ : *vrai*

● : *faux*

: portion explorée durant une résolution DFS à la volée

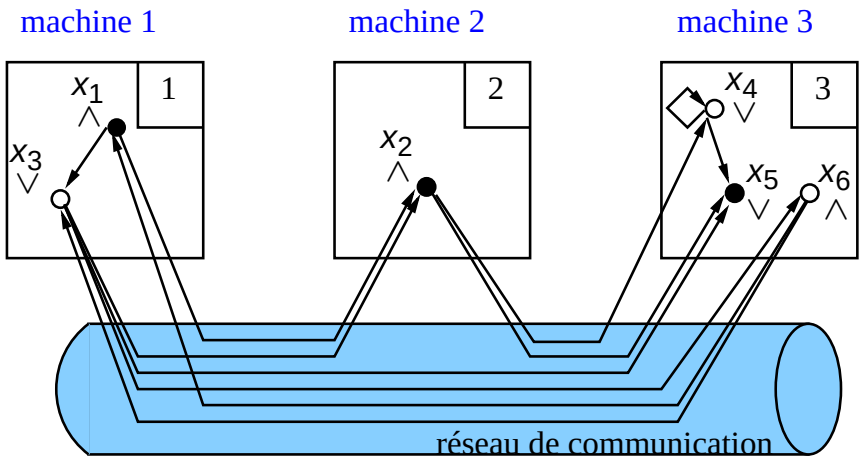
: diagnostic



- **Graphe booléen $G = (V, E, L)$ associé à un SEB (de signe ν)**
  - V = ensemble des variables
  - E = ensemble des arêtes
  - L = signe des variables ($\vee, \wedge$)
- **Résolution locale séquentielle [Mateescu-03]**
  - Valeur de vérité de la variable principale
  - Génération de diagnostic (sous-graphe booléen)

Distribution de la résolution de SEB

- Objectif : répartir le coût en mémoire sur plusieurs machines (limite actuelle 10^7 variables) et diminuer le temps de résolution (proportionnel à la taille du SEB)
- Méthode : distribution naturelle et équilibrée du problème de résolution de SEB par répartition des variables sur les différents processus



Plan de l'exposé

1. Systèmes d'Equations Booléennes

2. Résolution distribuée à la volée de SEBs
   - Résolution de SEB monobloc
   - Résolution de SEB multiblocs
   - Bibliothèque générique CAESAR_SOLVE_2

3. Trois applications en vérification énumérative

4. Application à la génération de tests

5. Conclusion et perspectives

Résolution de SEB monobloc

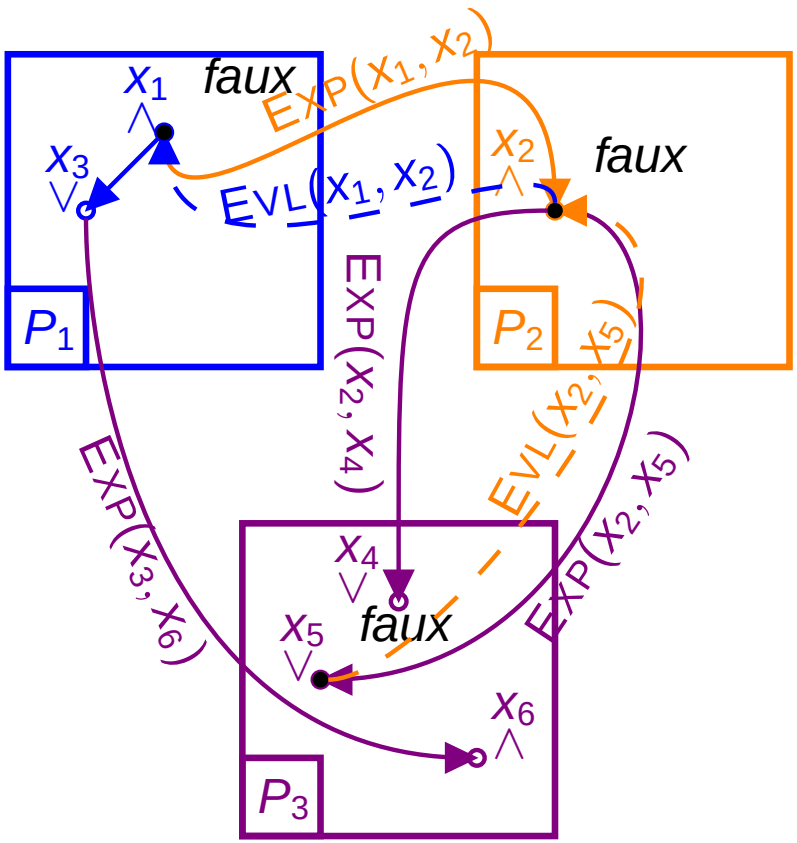
Modèle de calcul

- Architecture à mémoire distribuée (passage de messages) : grappe de PCs
- P processus SPMD et 1 processus superviseur
- Chaque processus résout un sous-ensemble des variables booléennes (fonction de hachage statique)

Algorithme distribué : DSOLVE

- Exploration en avant du graphe booléen (V, E, L) à partir de la variable principale $x \epsilon V$
- Propagation arrière de variables stables
- Distribution des variables à travers les dépendances distantes
- Détection de terminaison : x stable ou graphe booléen entièrement résolu

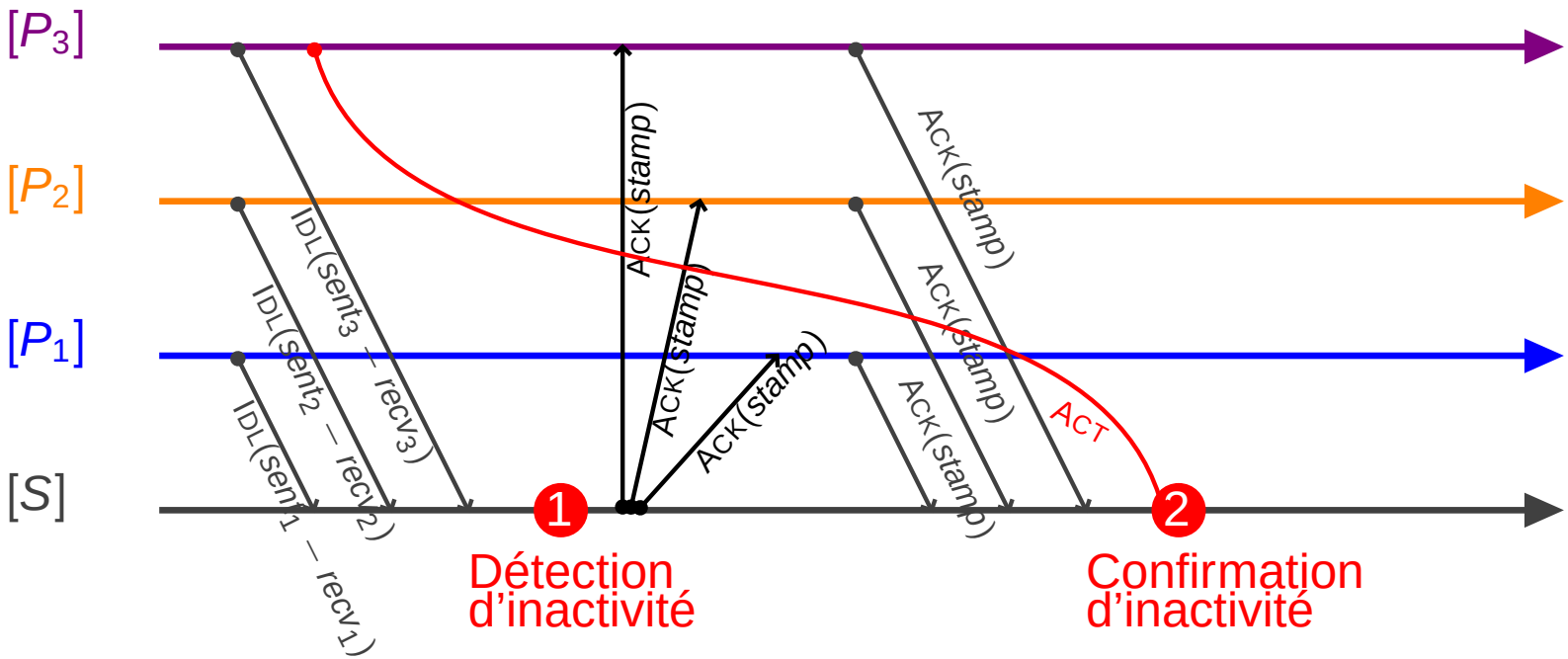
Exécution de DSolve



1. Initialisation (variable principale x_1)
2. Expansion locale et expansion distante (message Exp)
3. Variable conjonctive sans successeur (i.e., constante **fausse**)
4. Propagation arrière locale et distante (message Evl) de variables stabilisées (i.e., calculées)
5. Si la variable principale se stabilise, alors la résolution se termine

Algorithme de détection distribuée de terminaison (DDT)

- Deux vagues de détection d'inactivité globale entre le processus superviseur et les processus de résolution



Résultats de complexité

Pour un graphe booléen (V, E, L) et P processus de résolution :

- Complexité en temps dans le pire des cas = $O(|V| + |E|)$
  - deux parcours de graphe (avant et arrière) entrelacés
- Complexité en mémoire dans le pire des cas = $O(|V| + |E|)$
  - Dépendences stockées durant l'exploration du graphe
- Complexité en nombre de messages = $O(|E|)$
  - deux messages (expansion et stabilisation) sont au plus échangés par transition
- Détection distribuée de terminaison = $O(|E|)$
  - deux vagues avec $3P$ messages au plus échangés par transition

Résolution de SEB multiblocs

- Approche séquentielle [Mateescu-03] :
  - appels récursifs de résolution par bloc
  - pile d'appels bornée par le nombre de blocs du SEB
- Approche distribuée naïve (DSOLVE) :
  - une seule résolution pour le SEB entier
  - détection de terminaison du SEB entièrement résolu

→ incompatible ou inefficace avec résolution distribuée de SEB multiblocs

- Solution adoptée :
  - distinction des variables de blocs différents
  - détection distribuée de terminaison par bloc
  - gestion de deux parcours (avant et arrière) par bloc

Résolution distribuée de SEB multiblocs

- Extension conservative de l'algorithme DSOLVE $\Longrightarrow$ modèle de calcul identique

Algorithme distribué MB-DSOLVE

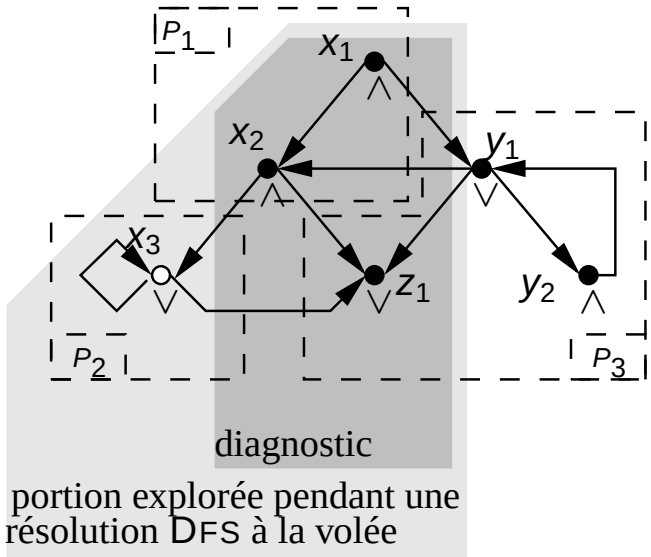
- Choix du numéro de bloc parmi ceux en attente d'exploration ou de stabilisation
- Priorité à la stabilisation de bloc de plus haut niveau dans l'arbre des dépendances entre blocs
- Limitation des requêtes d'exploration : une seule portion de bloc explorée à la fois, et priorité au bloc de plus bas niveau
- Gestion des transitions interblocs non stabilisées : propagations résiduelles
- Détection distribuée de portion de bloc résolu

Exemple de résolution distribuée à la volée de SEB multiblocs

bloc 1
$$x_1 \overset{\nu}{=} x_2 \wedge y_1$$
$$x_2 \overset{\nu}{=} x_3 \wedge z_1$$
$$x_3 \overset{\nu}{=} x_3 \vee z_1$$

bloc 2
$$y_1 \overset{\mu}{=} x_2 \vee x_{1,3} \vee y_2$$
$$y_2 \overset{\mu}{=} y_1$$

bloc 3
$$z_1 \overset{\nu}{=} \mathrm{F}$$



- Points fixes pouvant être différents entre les blocs
- Transitions interblocs à stabiliser

Bibliothèque générique CAESAR_SOLVE_2

- Résolution distribuée à la volée de SEBs sans alternance et génération distribuée à la volée de diagnostics (sous-graphes booléens)
  - SEB monobloc - DSOLVE (10 000 lignes de code C)
  - SEB multiblocs - MB-DSOLVE (7 000 lignes de code C supplémentaires)
- Testée avec un générateur (1000 lignes de code C) paramétré de SEBs aléatoires
- Connectée à une bibliothèque prototype et générique de communication à travers des sockets TCP/IP
- API de résolution booléenne générique et indépendante de l'application, donnée par la bibliothèque CÆSAR_SOLVE_1 [Mateescu-03]

Plan de l'exposé

1. Systèmes d'Equations Booléennes

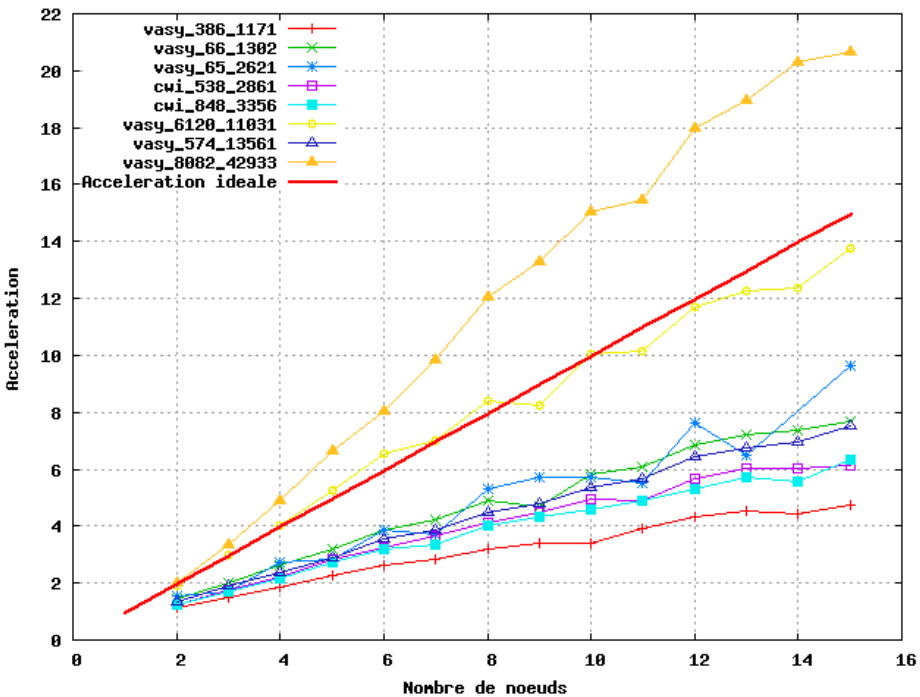
2. Résolution distribuée à la volée de SEBs

3. Trois applications en vérification énumérative
   - BISIMULATOR : comparateur à la volée par équivalences
   - TAU_CONFLUENCE : réducteur à la volée par tau-confluence
   - EVALUATOR 3.5 : évaluateur à la volée de formules logiques

4. Application à la génération de tests

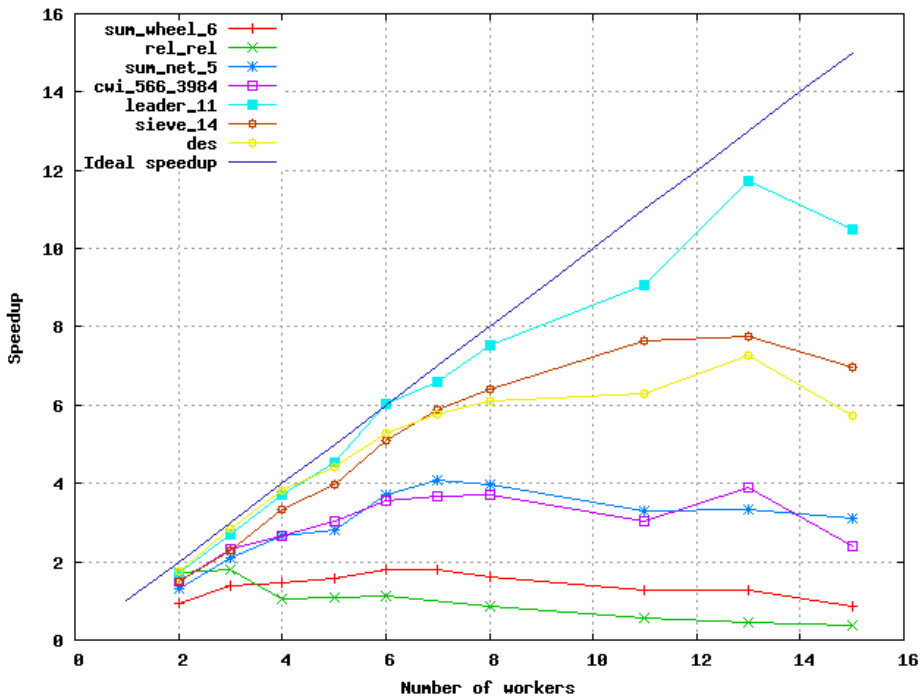
5. Conclusion et perspectives

BISIMULATOR distribué vs. séquentiel



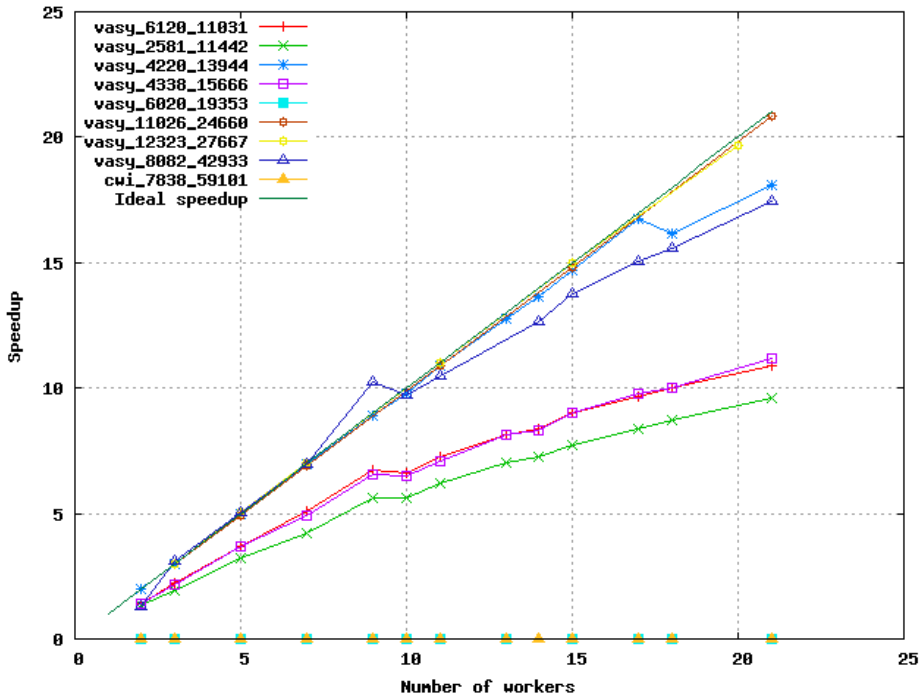
- Equivalence forte : meilleur comportement parmi toutes les équivalences (très peu de temps dans le calcul des successeurs)
- Accélérations linéaires
- vasy_6120_11031 (VLTS) :
  - 169.47 s. en séquentiel
  - 11.69 s. avec 15 processus, accélération de 14.5
- Surcoût mémoire constant (4 fois celui séquentiel)
  - quel que soit le nombre de nœuds de calcul
  - pour une taille de problèmes à résoudre

TAU_CONFLUENCE distribué vs. séquentiel



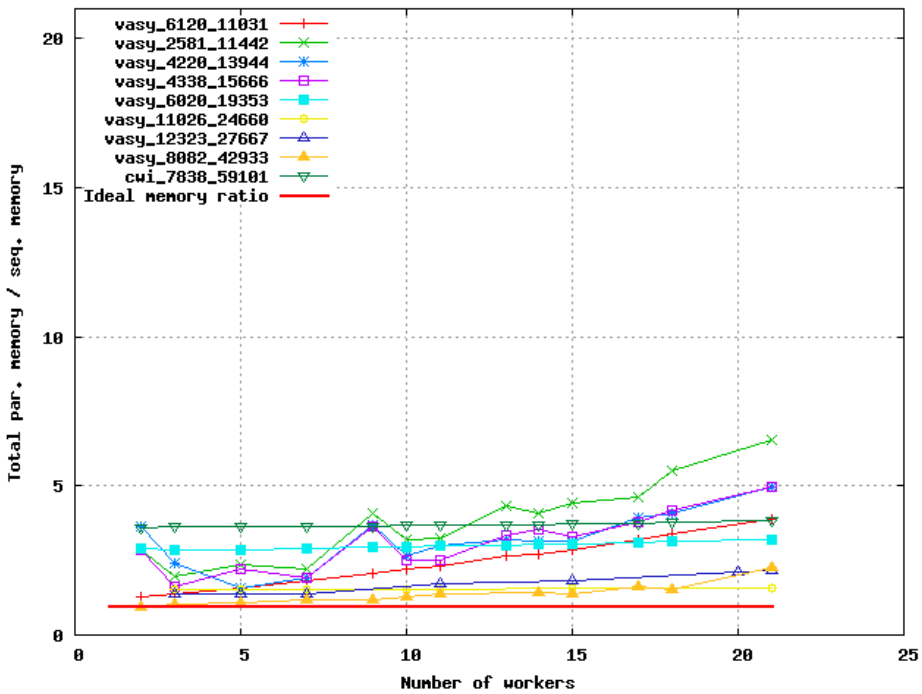
- Accélération proche du linéaire en nombre de noeuds utilisés
- Réduction entre un et quatre ordres de grandeur (comme en séquentiel)
- Limitations dans certains cas :
  - Parcours BFS avec appel de résolution pour chaque τ-transition
  - DDT forçant les noeuds à souvent se synchroniser
  - Solution alternative : appel sur un ensemble de τ-transitions
- Surcoût mémoire constant (3 fois celui séquentiel)
  - quel que soit le nombre de nœuds de calcul
  - très peu dépendant de la taille du problème

EVALUATOR 3.5 distribué vs. séquentiel (temps)



- Accélération proche du linéaire
- Comparable en temps et mémoire à UppDMC (model-checker distribué)
- Gain en temps significatif pour l'exemple *vasy_12323_27667* (VLTS) et la présence de famine :
  - $>$ 2 jours en séquentiel DFS optimisé
  - $<$ 3h en distribué sur 20 noeuds, accélération de 19.7
- Détection immédiate de diagnostics

EVALUATOR 3.5 distribué vs. séquentiel (mémoire)



- Surcoût mémoire constant (4 fois celui séquentiel) :
  - quel que soit le nombre de nœuds de calcul
  - pour une formule et sa valeur de vérité (détection de contre-exemples ou non)
- Evaluateur distribué pour d'autres logiques temporelles :
  - ACTL, par traduction en μ-calcul modal sans alternance [Fantechi-Gnesi-Ristori-92]

Plan de l'exposé

1. Systèmes d'Equations Booléennes

2. Résolution distribuée à la volée de SEBs

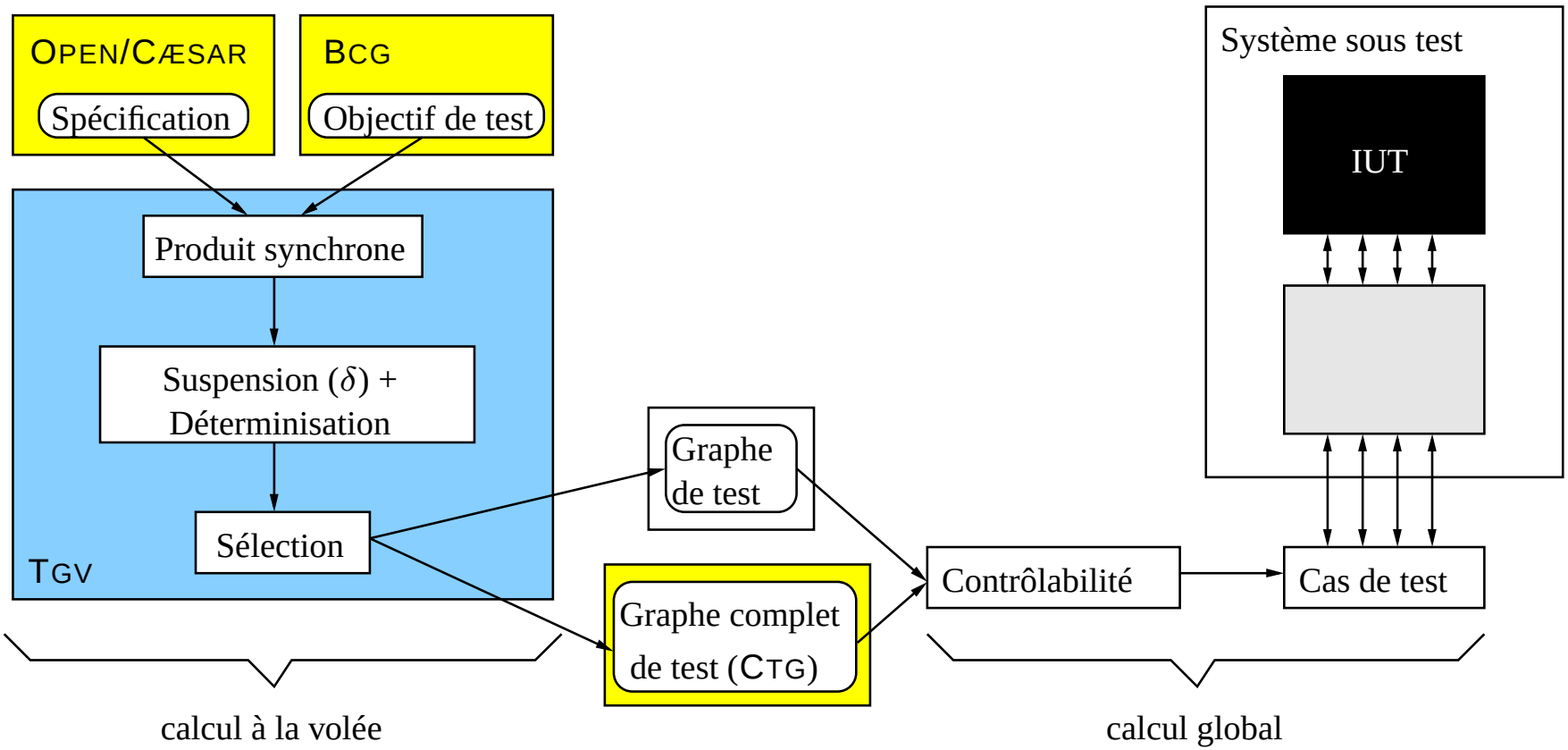
3. Trois applications en vérification énumérative

4. Application à la génération de tests
   - TGV : générateur à la volée de cas de tests
   - EXTRACTOR : générateur à la volée de cas de test

5. Conclusion et perspectives

TGV : générateur à la volée de cas de tests

- [Fernandez-Jard-Jeron-Viho-96], [Jard-Jeron-05]



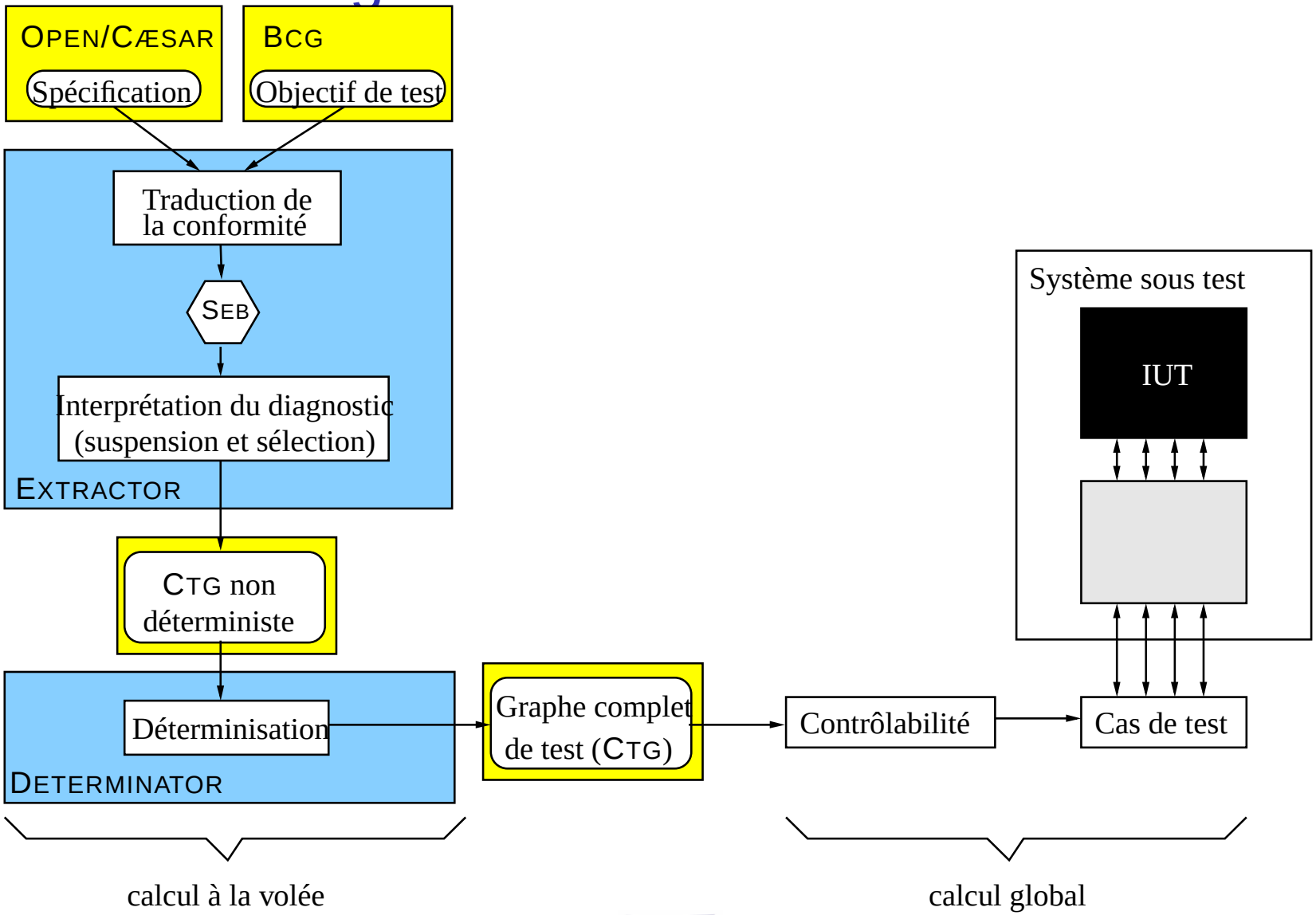
Traduction des cas de tests en termes de SEBs

- Génération de tests =
  - cas particulier de génération de diagnostic pour une formule de μ-calcul modal sans alternance
  - cas particulier de génération de diagnostic pour un SEB multiblocs
- Définition du SEB multiblocs correspondant :

$$\begin{array}{lll} \{X_s & =_\nu & Y_s \wedge \bigwedge_{s \to s'} (Z_{s'} \vee X_{s'})\} \\ \{Y_s & =_\mu & \bigvee_{s \xrightarrow{acc} s'} \mathsf{T} \vee \bigvee_{s \to s'} Y_{s'}\} \\ \{Z_s & =_\nu & \bigwedge_{s \xrightarrow{acc} s'} \mathsf{F} \wedge \bigwedge_{s \to s'} Z_{s'}\} \end{array}$$

- Avantages :
  - solution générique
  - obtention directe d'un générateur distribué à la volée de cas de tests

EXTRACTOR : générateur à la volée de cas de tests



EXTRACTOR vs. TGV

- Accélération :

$$\frac{\sum_{\text{STEs}} temps(\text{TGV})}{(\sum_{\text{STEs}} temps(\text{EXTRACTOR}) + \sum_{\text{CTGs } interm.} temps(\text{DETERMINATOR}))} = 1.82$$

- Consommation mémoire :

$$\frac{\sum_{\text{STEs}} memoire(\text{TGV})}{(\sum_{\text{STEs}} memoire(\text{EXTRACTOR}) + \sum_{\text{CTGs } interm.} memoire(\text{DETERMINATOR}))} = 1.05$$

- Taille des CTGs :

$$\frac{\sum_{\text{STEs}} nombreEtats(\text{TGV})}{\sum_{\text{CTGs } interm.} nombreEtats(\text{DETERMINATOR})} = 0.71$$

$$\frac{\sum_{\text{STEs}} nombreTrans(\text{TGV})}{\sum_{\text{CTGs } interm.} nombreTrans(\text{DETERMINATOR})} = 0.53$$

- Exemples traités sur lesquels TGV échoue :

EXEMPLE	M états	M trans.	EXTRACTOR + DETERMINATOR
cwi_214_684	214	684	8 s., 19 Mo, aucun cas de test
cwi_566_3984	566	3 984	1195 s., 145 Mo, (32 états, 49 trans.)

Plan de l'exposé

1. Systèmes d'Equations Booléennes
2. Résolution distribuée à la volée de SEBs
3. Trois applications en vérification énumérative
4. Application à la génération de tests
5. Conclusion et perspectives
   - Bilan
   - Perspectives

Bilan

1. Moteur générique de vérification distribuée à la volée :
   - Résolution de SEB monobloc (DSOLVE)
   - Résolution de SEB multiblocs (MB-DSOLVE)
2. Connexion à des outils réels de vérification formelle :
   - Comparaison à la volée par équivalences (BISIMULATOR)
   - Réduction à la volée par ordre partiel (TAU_CONFLUENCE)
   - Evaluation à la volée de formules de logiques temporelles (EVALUATOR)
3. Application à la génération de tests :
   - Traduction de la génération à la volée de cas de test de conformité en termes de SEBS (EXTRACTOR)
4. Expérimentation massive des outils sur cas d'études industrielles et machines parallèles réelles

Perspectives

- Compléter les applications existantes :
  - Traduction d'autres équivalences : bisimulation Markovienne [Hermanns-Siegle-99]
  - Traduction d'autres réductions : tau-inertness [Groote-Sellink-90], tau-confluence faible [Groote-vandePol-00]
- Développer d'autres applications à partir de DSOLVE et MB-DSOLVE :
  - Résolution de clauses de Horn `[Liu-Smolka-98]`
  - Analyse de flot de données et interprétation abstraite `[Fecht-Seidl-96]`
- Etudier d'autres stratégies de résolution de SEBS
- Généraliser l'approche à des architectures hétérogènes, telles que les NOWS, puis les grilles de calcul

Bibliographie


C. Joubert et R. Mateescu.
Distributed On-the-Fly Equivalence Checking and τ-Confluence Reduction.
FMSD'2006, en cours de publication.


D. Bergamini, N. Descoubes, C. Joubert et R. Mateescu.
BISIMULATOR : A Modular Tool for On-the-Fly Equivalence Checking.
TACAS'2005, LNCS 3440 :581–585.


C. Joubert et R. Mateescu.
Distributed Local Resolution of Boolean Equation Systems.
PDP'2005, IEEE 264–271.


C. Joubert et R. Mateescu.
Distributed On-the-Fly Equivalence Checking.
PDMC'2004, ENTCS 128(3).


Christophe Joubert.
Distributed Model-Checking : From Abstract Algorithms to Concrete Implementations.
PDMC'2003, ENTCS 89(1).


H. Hermanns et C. Joubert.
A Set of Performance and Dependability Analysis Components for CADP.
TACAS'2003, LNCS 2619 :425–430.