Contributions à l'approche automate pour la vérification de propriétés de systèmes concurrents

Alexandre Duret-Lutz

10 juillet 2007

M. **Ahmed Bouajjani**, Pr. à l'Université Paris 7	Rapporteur
M. **François Vernadat**, Pr. à l'INSA de Toulouse	Rapporteur
M. **Jean-Michel Couvreur**, Pr. à l'Université d'Orléans	Examinateur
M. **Claude Girault**, Pr. émérite à l'Université Paris 6	Examinateur
M. **Alain Griffault**, M.d.C. à l'Université Bordeaux 1	Examinateur
M. **Serge Haddad**, Pr. à l'Université Paris-Dauphine	Examinateur
M. **Fabrice Kordon**, Pr. à l'Université Paris 6	Directeur de thèse
M. **Denis Poitrenaud**, M.d.C. à l'Université Paris 5	Encadrant

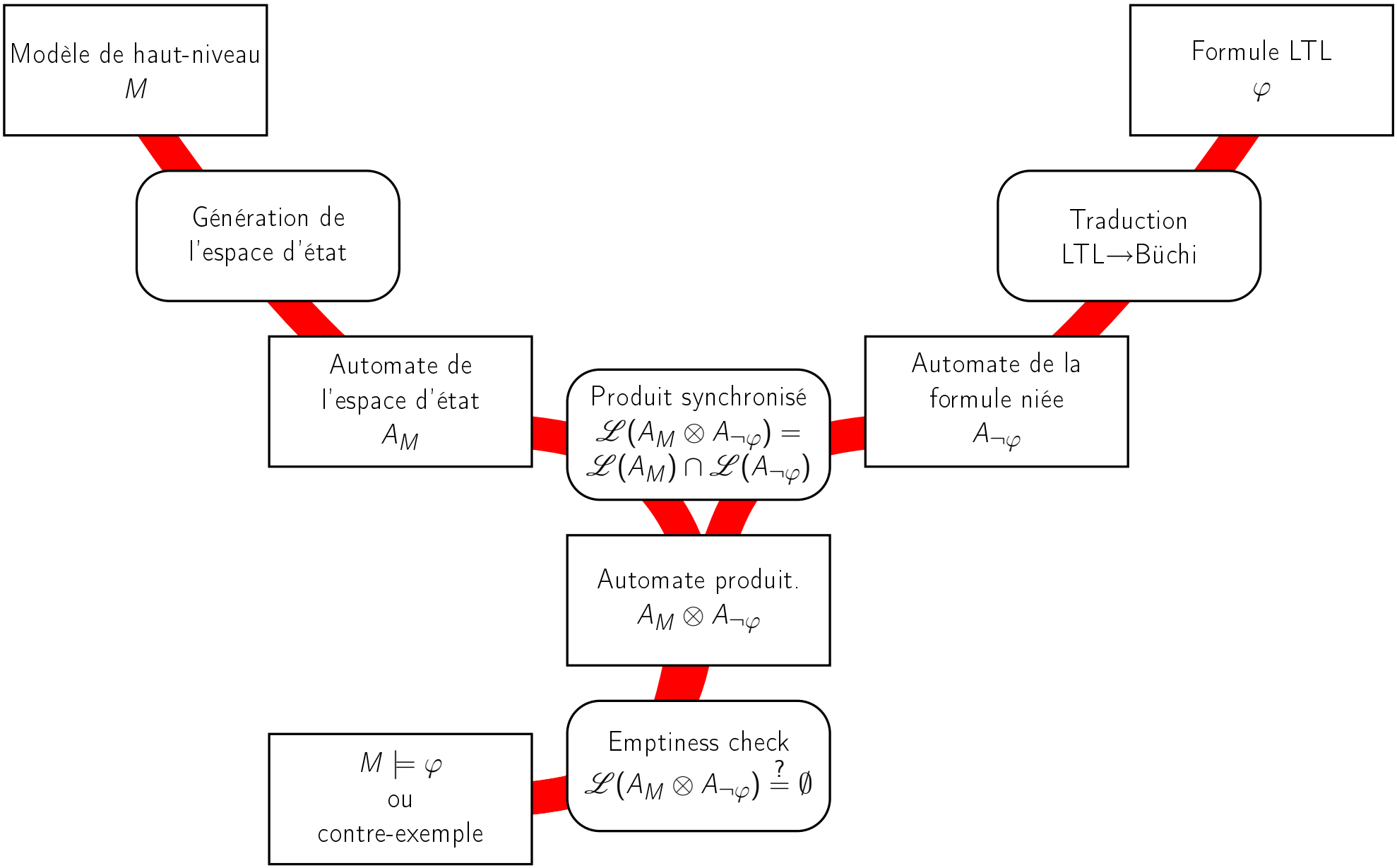
Contexte : vérification formelle

Vérification formelle :
(outils mathématiques pour garantir la correction d'un **modèle**)

- preuves de théorèmes
  - plus ou moins manuelles...
- *model checking* = approche algorithmique
  - automatique
  - plus contrainte (p.ex., nombre fini d'états)
  - *model checkers* utilisés par AT&T, Cadence, Fujitsu, HP, IBM, Intel, Motorola, NEC, SGI, Siemens, Sun...

Nous nous intéressons à l'approche automate du *model checking*.

Approche automate du *model checking*



Sur l'approche automate

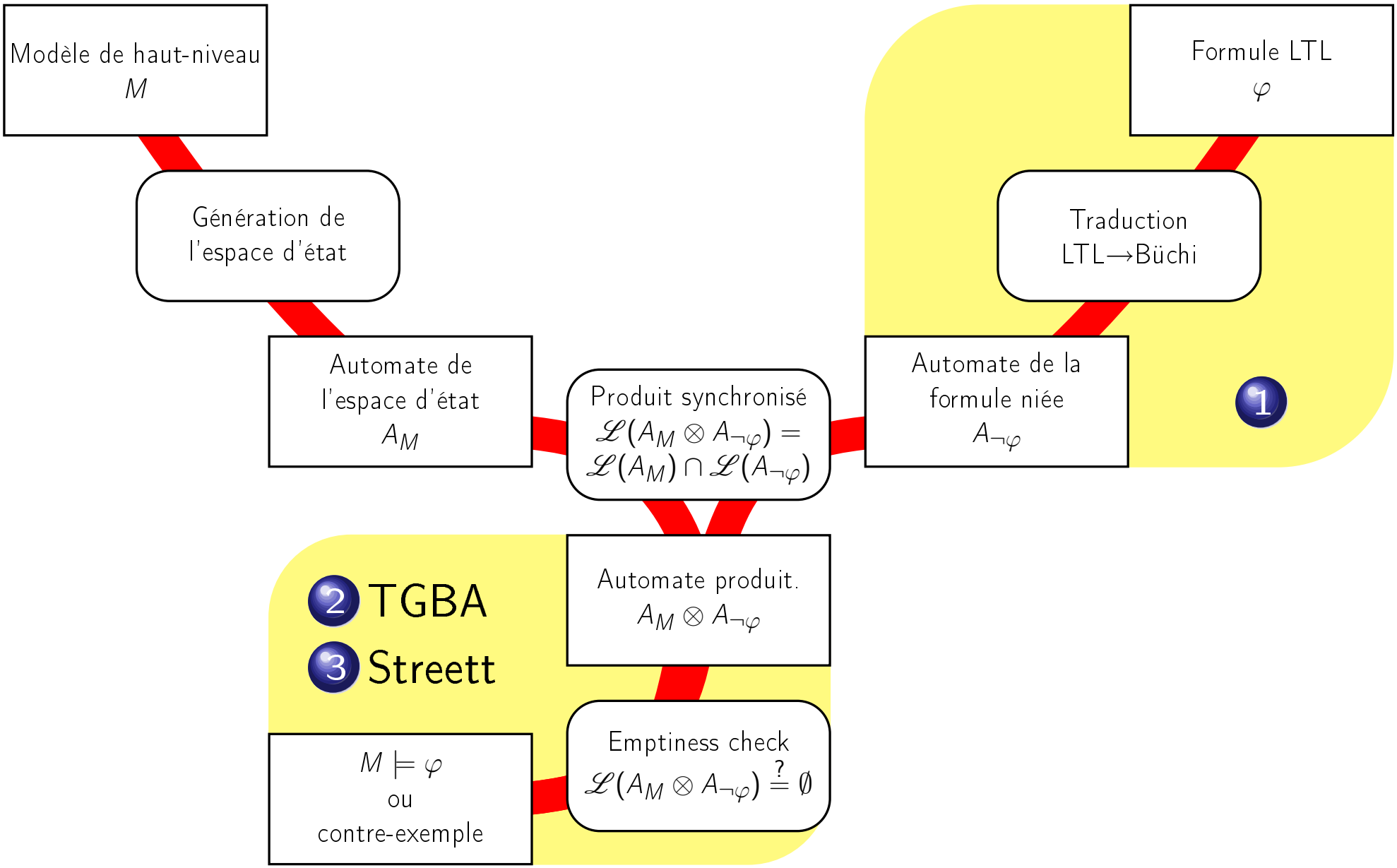
Problématique générale :

- *Size matters*
- On chercher à réduire la taille des automates manipulés

Remarques :

- Domaine de recherche vieux de 20 ans [Vardi & Wolper '86]
- Mes apports portent sur différents algorithmes à différentes étapes de l'approche
  $\Rightarrow$ Impossible de tout présenter en contexte
- Ce travail promeut les automates étiquetés sur les transitions

Approche automate du *model checking*



Traduction de formules LTL en automates de Büchi

Je vais présenter une méthode combinant plusieurs notions existantes :

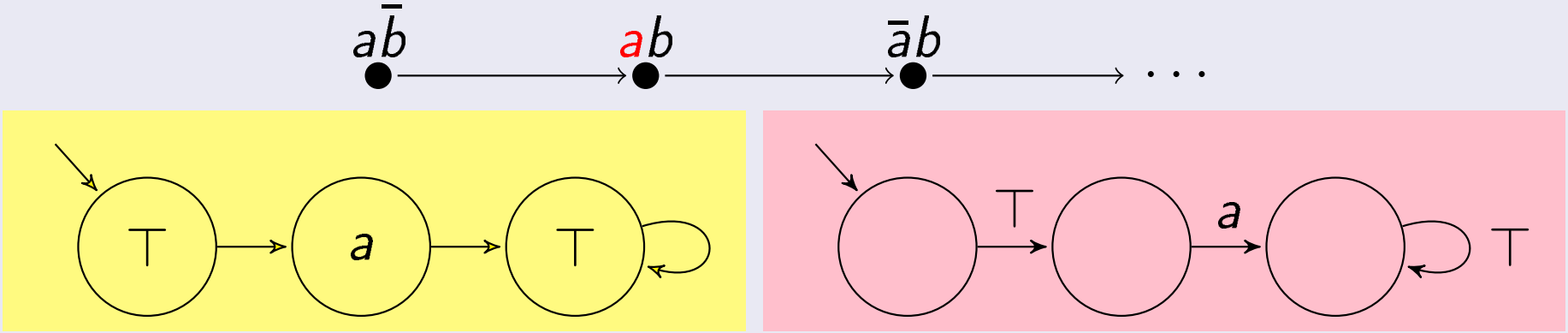
- un tableau présenté de façon arborescente [Wolper '83]
- la notion de promesse [Haddad & Vernadat '03]
- la prise en compte de ces promesses comme des termes dans le tableau [Couvreur '99]

La nouvelle présentation obtenue

- est simple à expliquer ;
- permet de construire automates sur états ou transitions ;
- montre que les automates sur états sont forcément plus gros que ceux sur transitions.

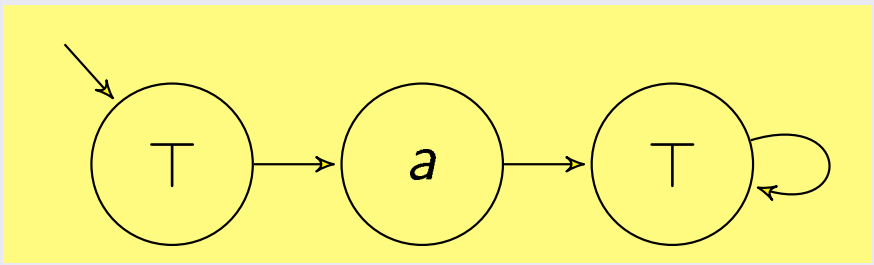
LTL et automates sur états ou transitions

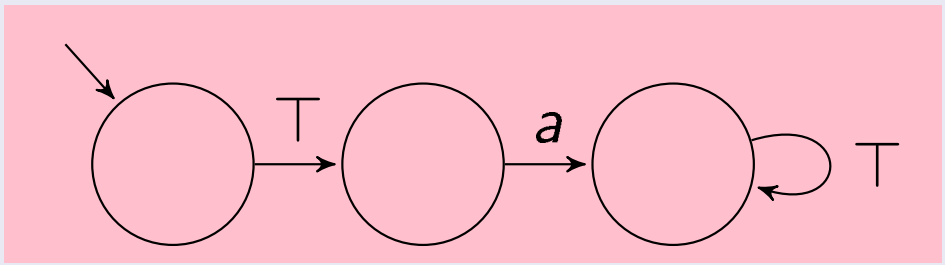
X a



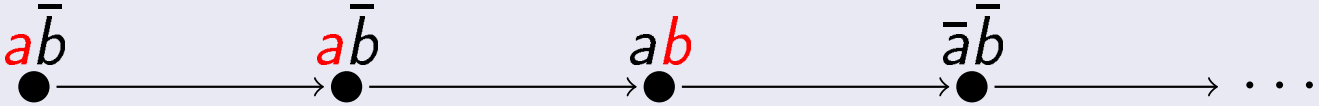
LTL et automates sur états ou transitions

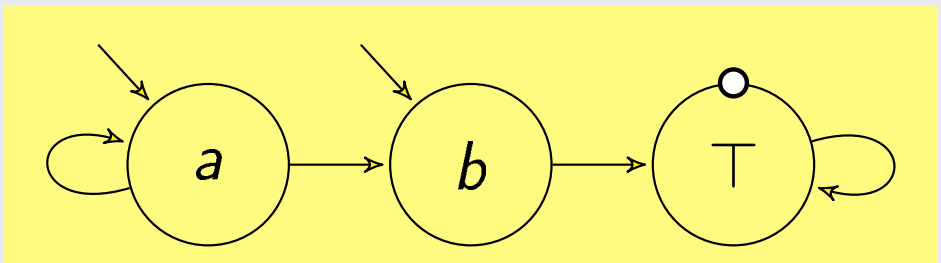
X a

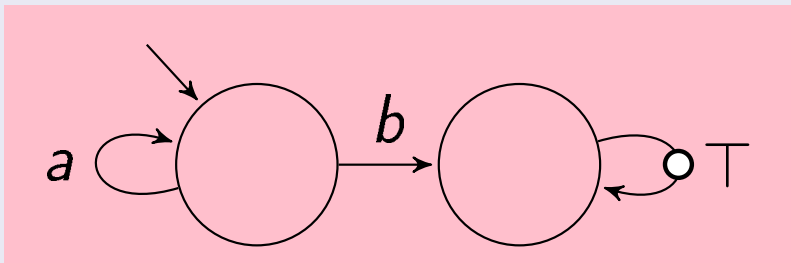




$a \mathsf{U} b$







$$a \mathsf{U} b \equiv b \vee (a \wedge \mathsf{X}(a \mathsf{U} b))$$

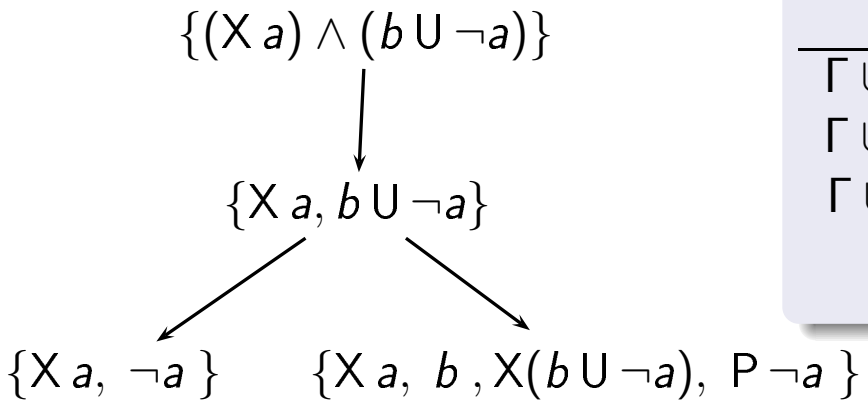
Tableau pour $(\mathsf{X}\, a) \wedge (b\, \mathsf{U}\, \neg a)$

$\{(\mathsf{X}\, a) \wedge (b\, \mathsf{U}\, \neg a)\}$

Règles de tableau

formule	1^{er} fils	2^{e} fils
$\Gamma \cup \{f \wedge g\}$	$\Gamma \cup \{f, g\}$	
$\Gamma \cup \{f \vee g\}$	$\Gamma \cup \{f\}$	$\Gamma \cup \{g\}$
$\Gamma \cup \{f\, \mathsf{U}\, g\}$	$\Gamma \cup \{g\}$	$\Gamma \cup \{f, \mathsf{X}(f\, \mathsf{U}\, g), \mathsf{P}\, g\}$
$\vdots$	$\vdots$	$\vdots$

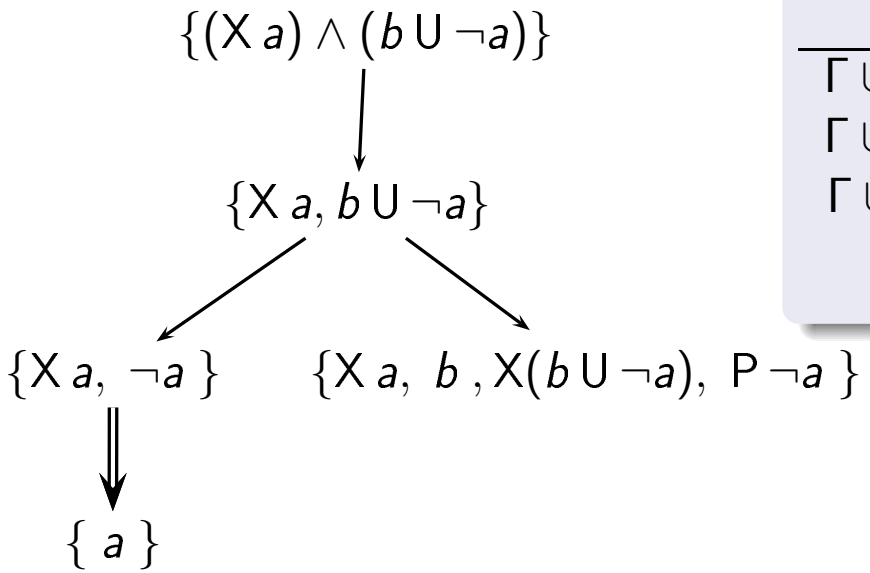
Tableau pour $(\mathsf{X}\, a) \wedge (b\, \mathsf{U}\, \neg a)$



Règles de tableau

formule	1^{er} fils	2^{e} fils
$\Gamma \cup \{f \wedge g\}$	$\Gamma \cup \{f, g\}$	
$\Gamma \cup \{f \vee g\}$	$\Gamma \cup \{f\}$	$\Gamma \cup \{g\}$
$\Gamma \cup \{f\, \mathsf{U}\, g\}$	$\Gamma \cup \{g\}$	$\Gamma \cup \{f, \mathsf{X}(f\, \mathsf{U}\, g), \mathsf{P}\, g\}$
$\vdots$	$\vdots$	$\vdots$

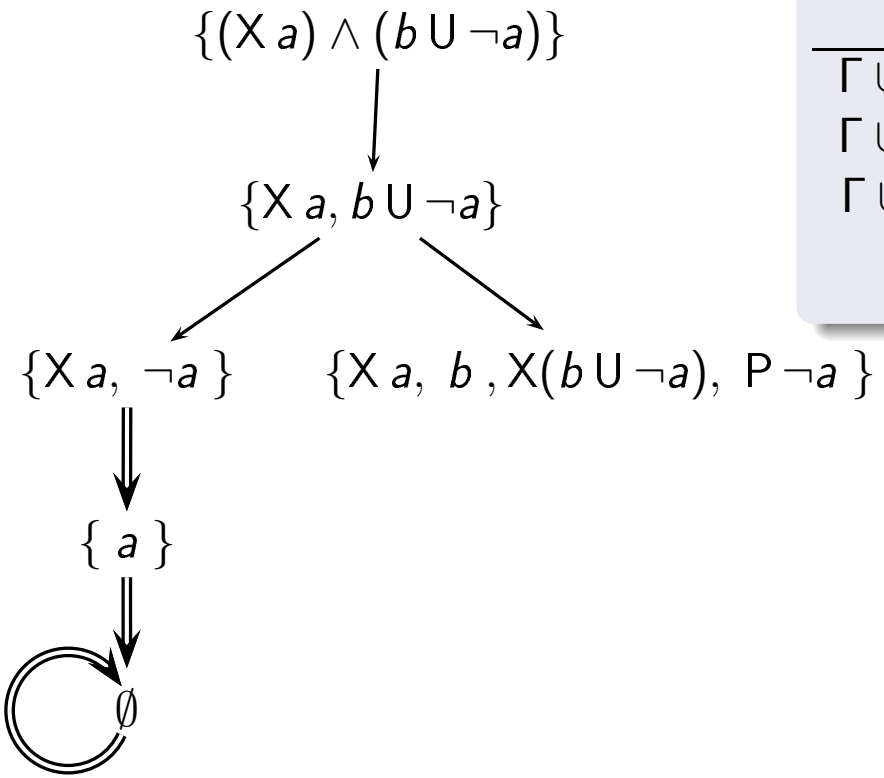
Tableau pour $(\mathsf{X}\, a) \wedge (b\, \mathsf{U}\, \neg a)$



Règles de tableau

formule	1^{er} fils	2^{e} fils
$\Gamma \cup \{f \wedge g\}$	$\Gamma \cup \{f, g\}$	
$\Gamma \cup \{f \vee g\}$	$\Gamma \cup \{f\}$	$\Gamma \cup \{g\}$
$\Gamma \cup \{f\, \mathsf{U}\, g\}$	$\Gamma \cup \{g\}$	$\Gamma \cup \{f, \mathsf{X}(f\, \mathsf{U}\, g), \mathsf{P}\, g\}$
$\vdots$	$\vdots$	$\vdots$

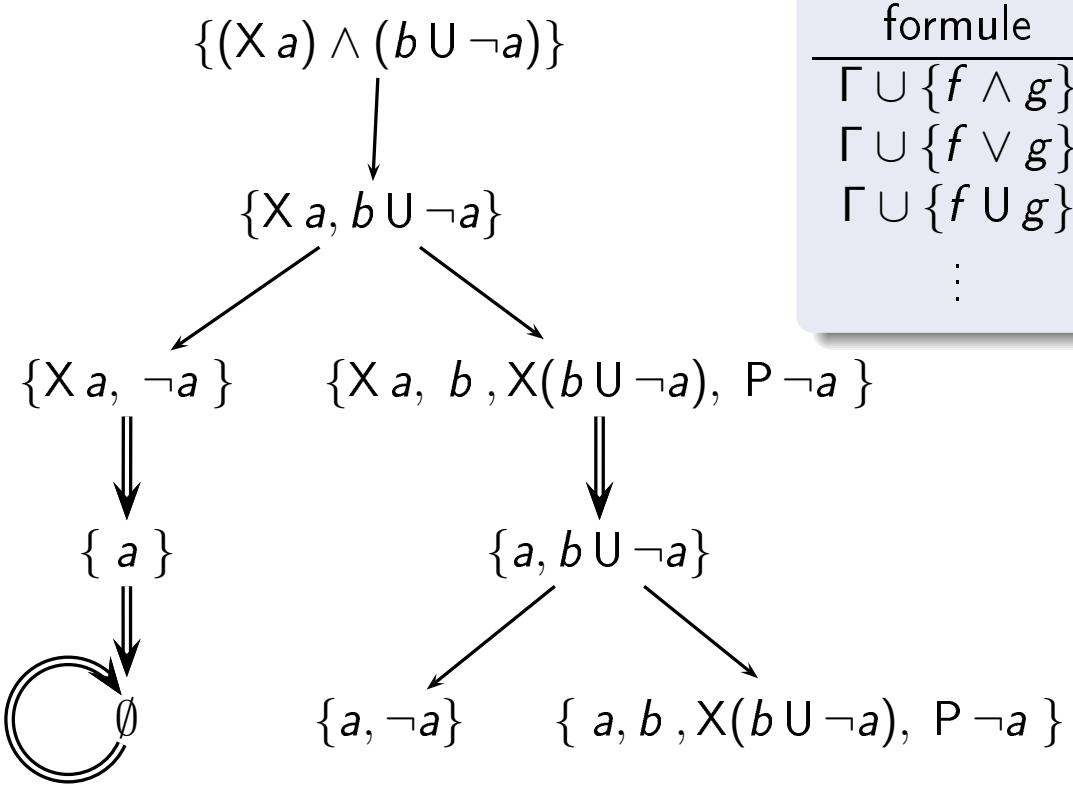
Tableau pour $(\mathsf{X}\, a) \wedge (b\, \mathsf{U}\, \neg a)$



Règles de tableau

formule	1^{er} fils	2^{e} fils
$\Gamma \cup \{f \wedge g\}$	$\Gamma \cup \{f, g\}$	
$\Gamma \cup \{f \vee g\}$	$\Gamma \cup \{f\}$	$\Gamma \cup \{g\}$
$\Gamma \cup \{f\, \mathsf{U}\, g\}$	$\Gamma \cup \{g\}$	$\Gamma \cup \{f, \mathsf{X}(f\, \mathsf{U}\, g), \mathsf{P}\, g\}$
$\vdots$	$\vdots$	$\vdots$

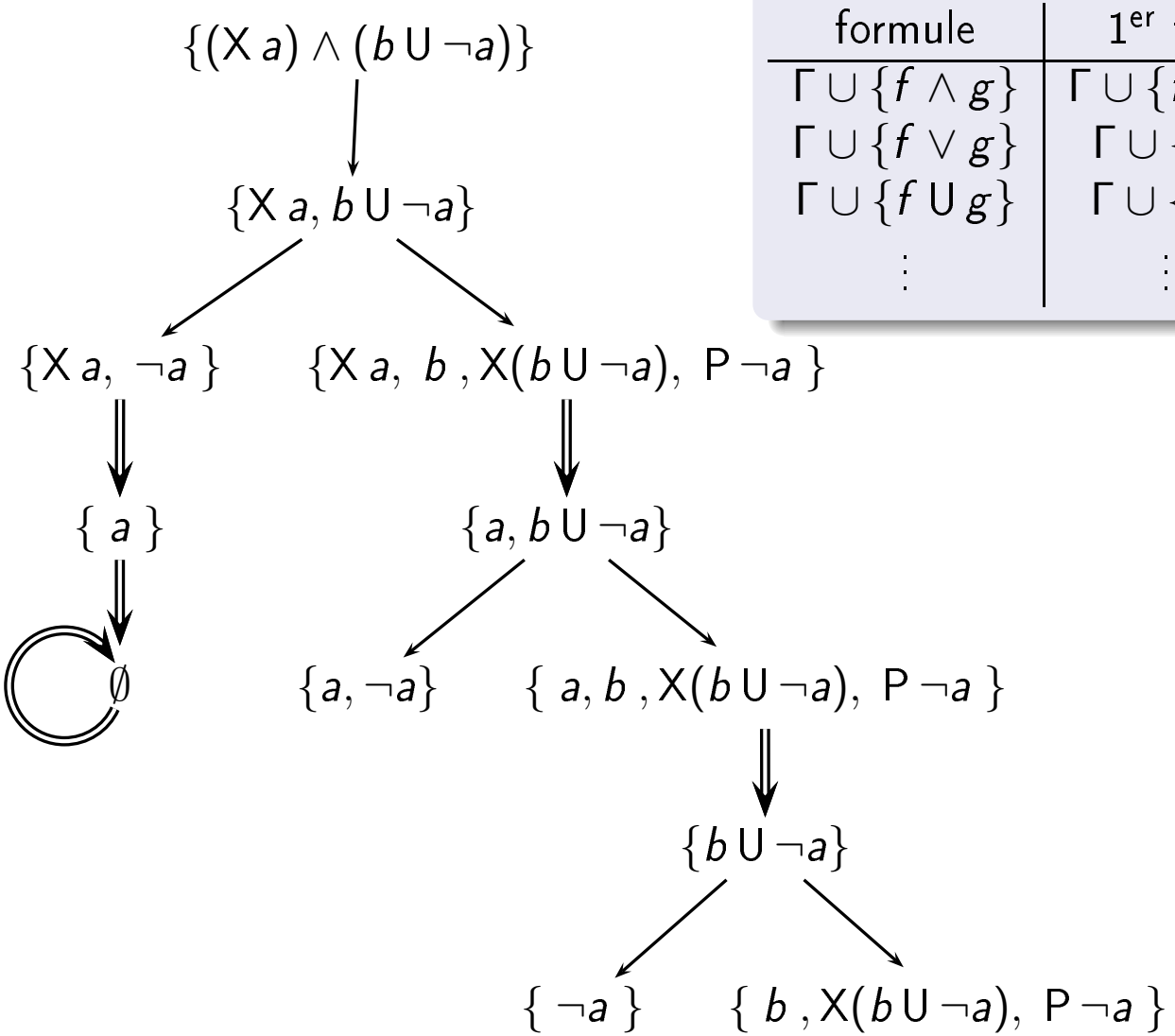
Tableau pour $(\mathsf{X}\, a) \wedge (b\, \mathsf{U}\, \neg a)$



Règles de tableau

formule	1^{er} fils	2^{e} fils
$\Gamma \cup \{f \wedge g\}$	$\Gamma \cup \{f, g\}$	
$\Gamma \cup \{f \vee g\}$	$\Gamma \cup \{f\}$	$\Gamma \cup \{g\}$
$\Gamma \cup \{f\, \mathsf{U}\, g\}$	$\Gamma \cup \{g\}$	$\Gamma \cup \{f, \mathsf{X}(f\, \mathsf{U}\, g), \mathsf{P}\, g\}$
$\vdots$	$\vdots$	$\vdots$

Tableau pour $(\mathrm{X}\, a) \wedge (b \,\mathrm{U}\, \neg a)$



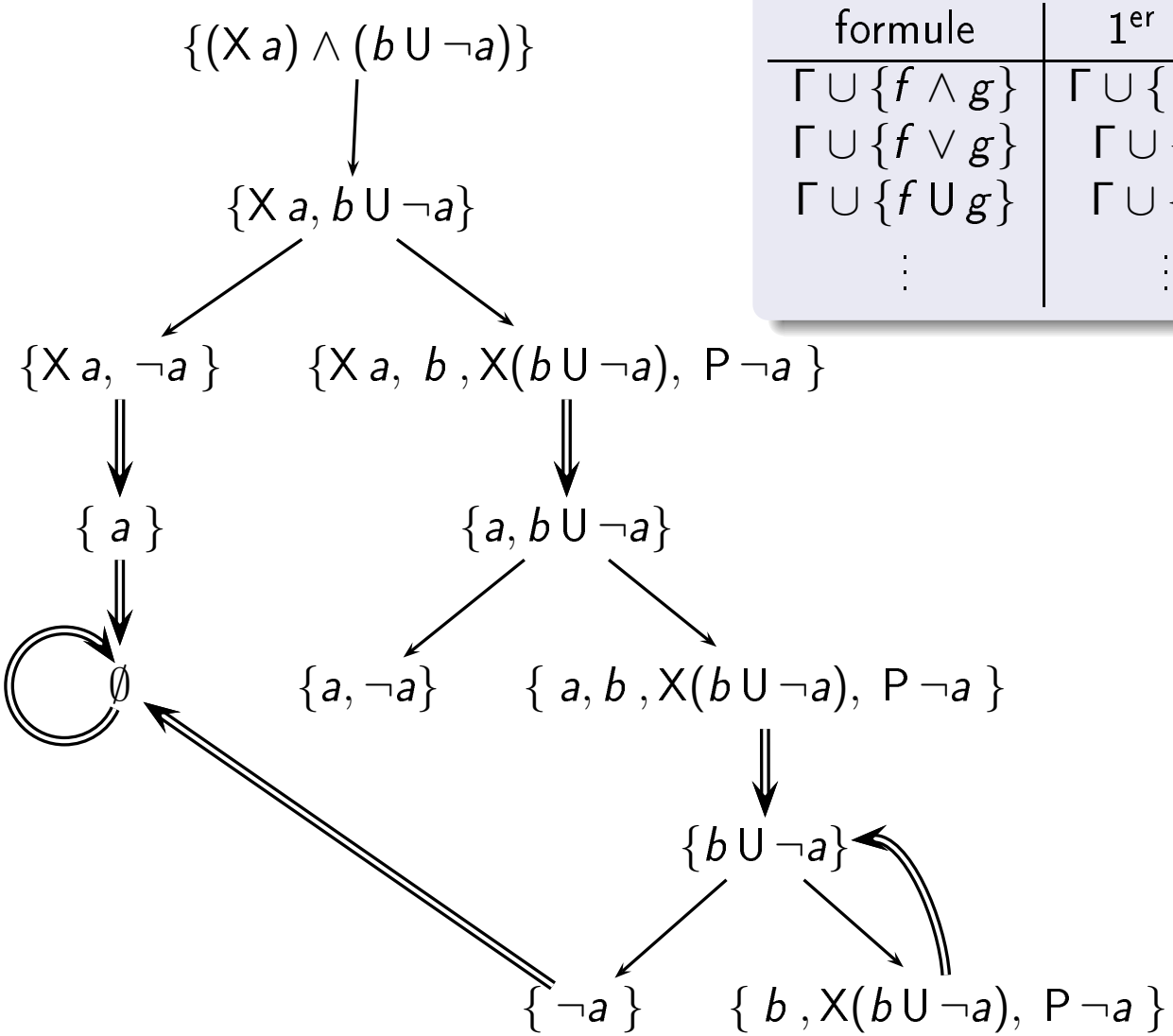
Règles de tableau

formule	1er fils	2e fils
$\Gamma \cup \{f \wedge g\}$	$\Gamma \cup \{f, g\}$	
$\Gamma \cup \{f \vee g\}$	$\Gamma \cup \{f\}$	$\Gamma \cup \{g\}$
$\Gamma \cup \{f \,\mathrm{U}\, g\}$	$\Gamma \cup \{g\}$	$\Gamma \cup \{f, \mathrm{X}(f \,\mathrm{U}\, g), \mathrm{P}\, g\}$
⋮	⋮	⋮

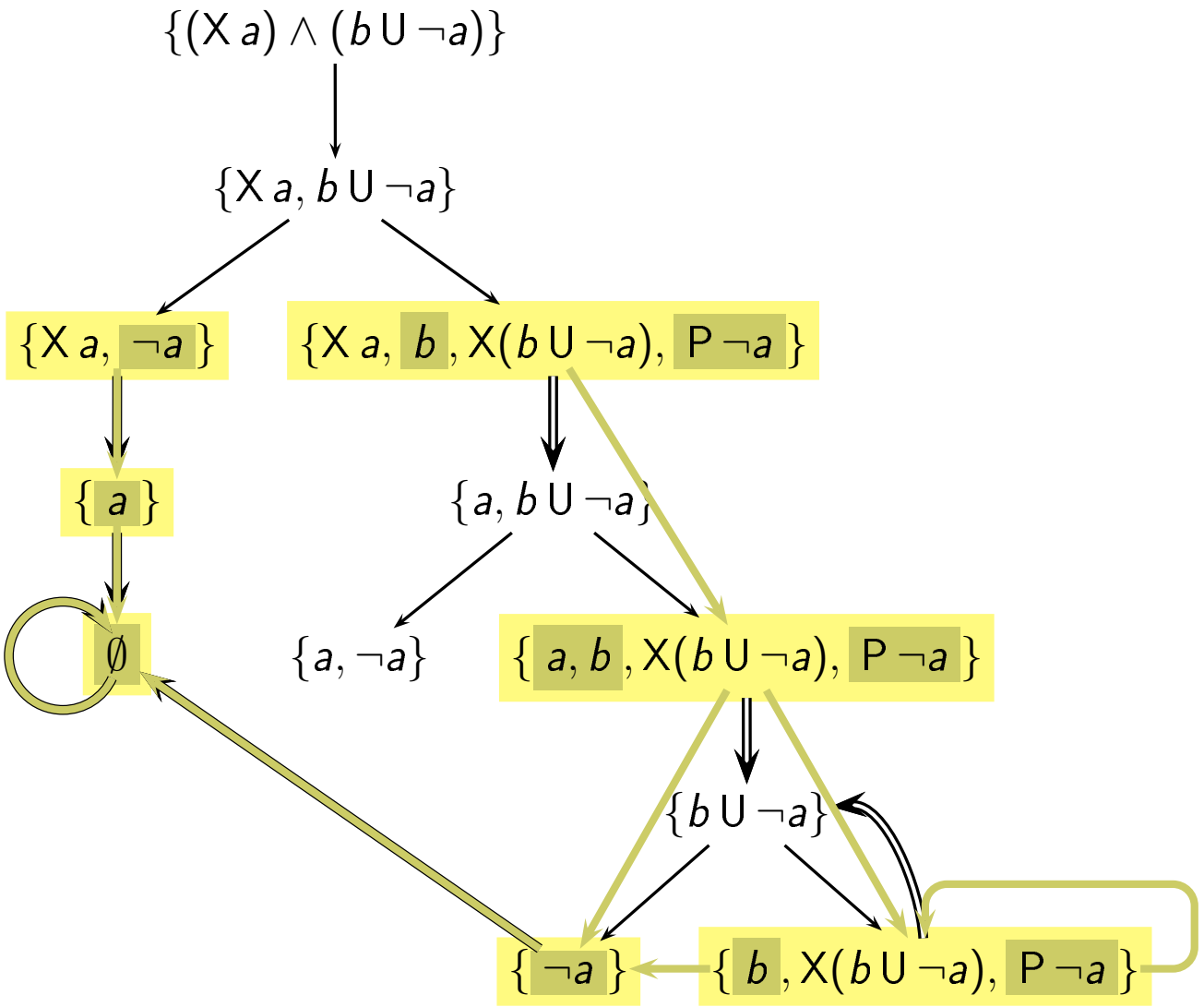
Tableau pour $(\mathrm{X}\, a) \wedge (b \,\mathrm{U}\, \neg a)$

Règles de tableau

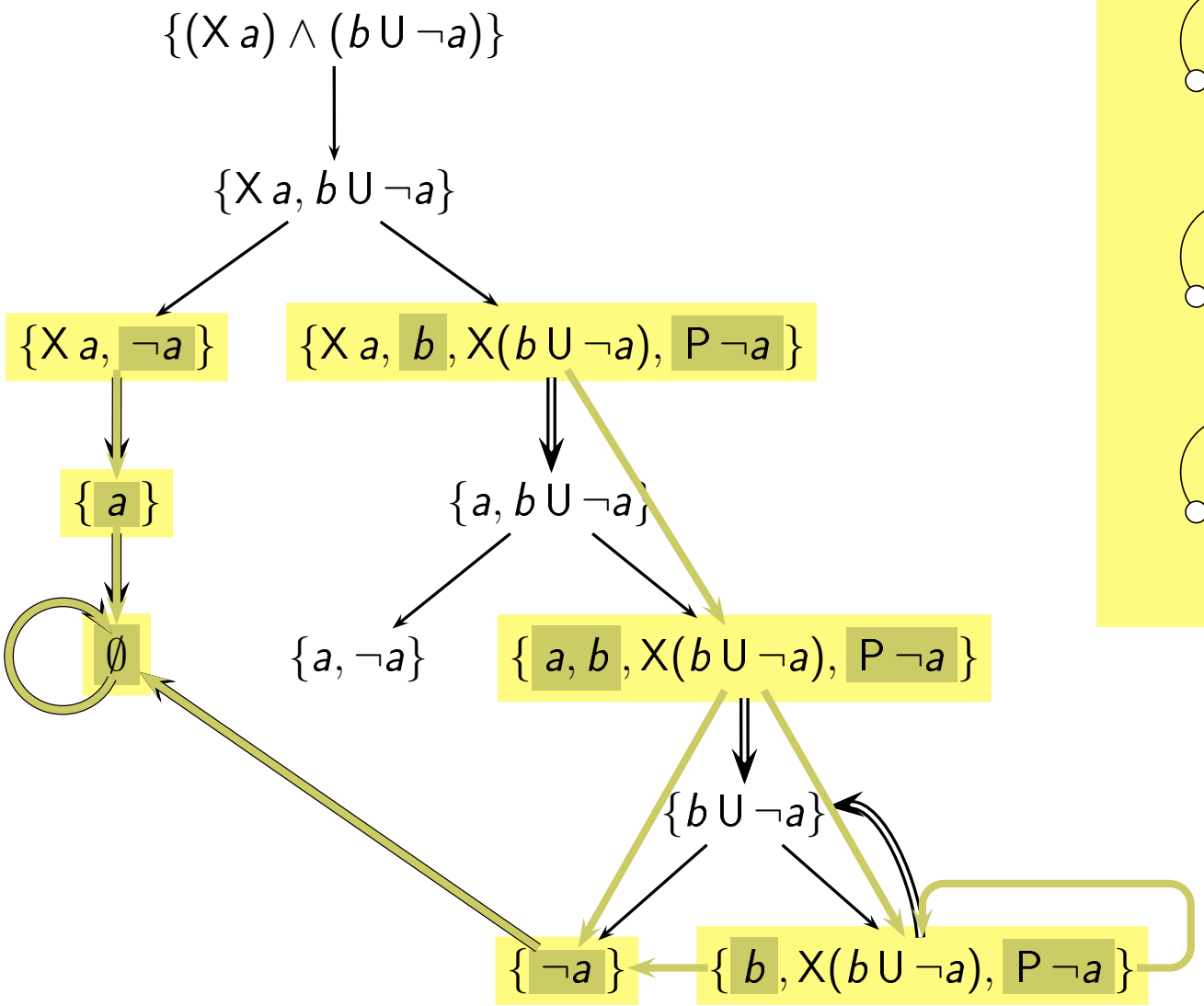
formule	1^{er} fils	2^{e} fils
$\Gamma \cup \{f \wedge g\}$	$\Gamma \cup \{f, g\}$	
$\Gamma \cup \{f \vee g\}$	$\Gamma \cup \{f\}$	$\Gamma \cup \{g\}$
$\Gamma \cup \{f \,\mathrm{U}\, g\}$	$\Gamma \cup \{g\}$	$\Gamma \cup \{f, \mathrm{X}(f \,\mathrm{U}\, g), \mathrm{P}\, g\}$
$\vdots$	$\vdots$	$\vdots$

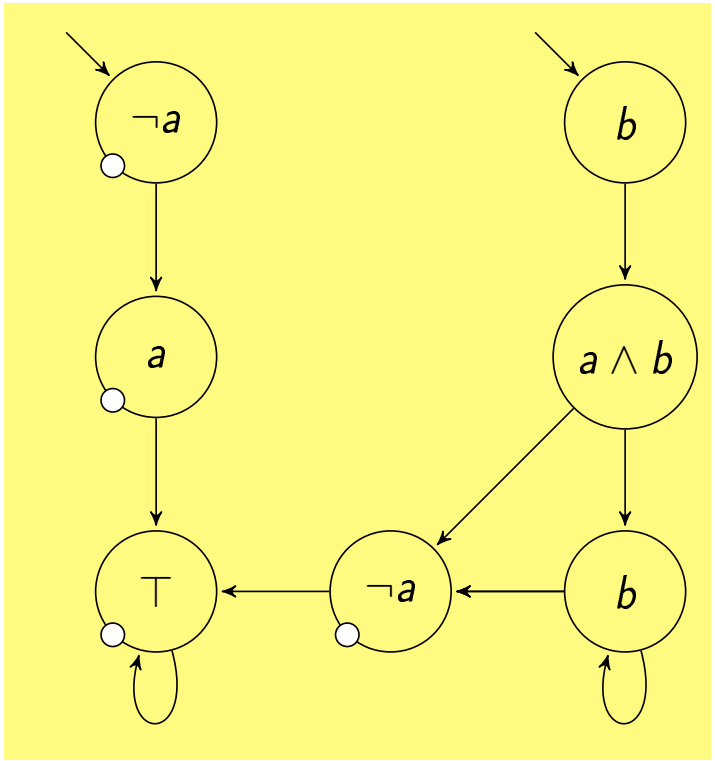


$(\mathrm{X}\, a) \wedge (b\, \mathrm{U}\, \neg a)$ vers GBA basés sur les états

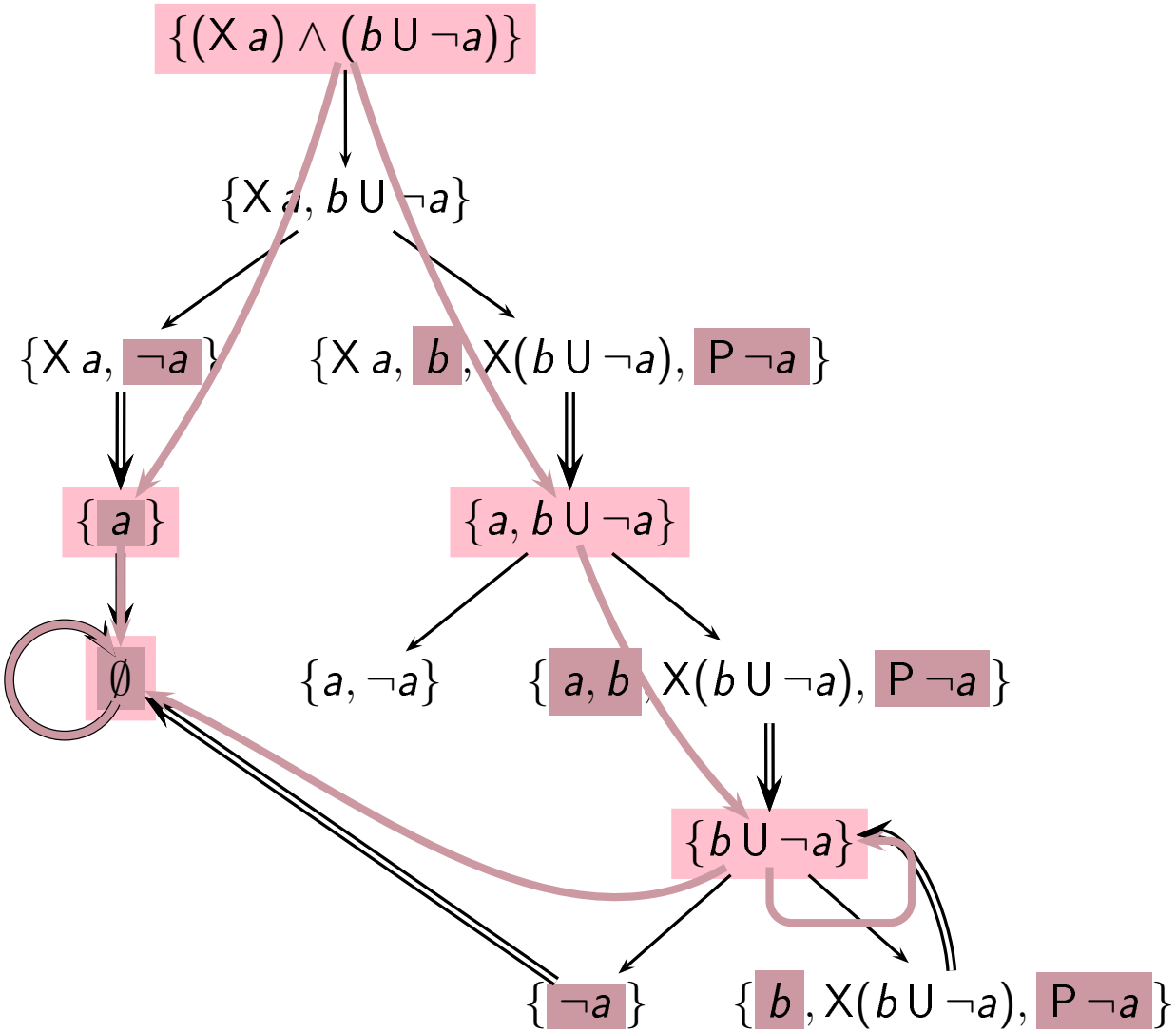


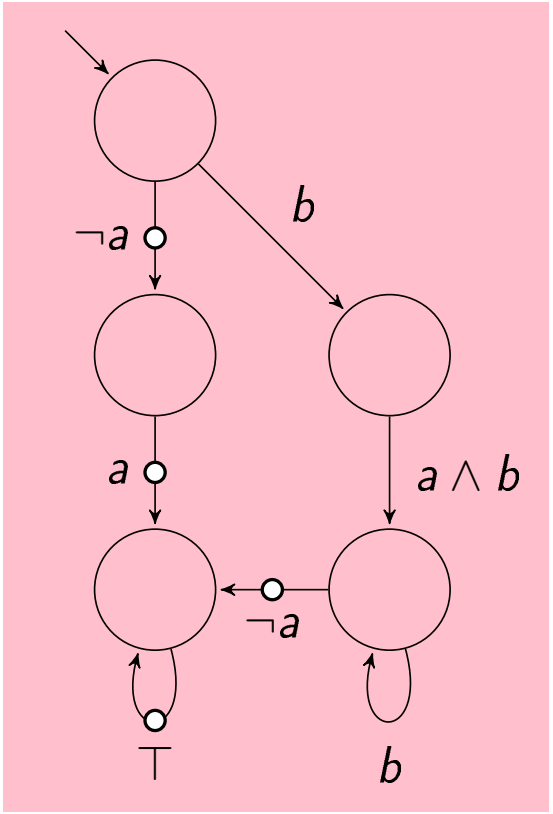
$(\mathrm{X}\, a) \wedge (b\, \mathrm{U}\, \neg a)$ vers GBA basés sur les états





$(\mathrm{X}\, a) \wedge (b\, \mathrm{U}\, \neg a)$ vers GBA basé sur les transitions





Contributions sur la traduction (1/3)

Présentation graphique des tableaux avec utilisation des promesses [MASCOTS'04]

- Simple à comprendre et expliquer
- Montre bien l'avantage des TGBA sur les GBA
- La version sur états comparable à une version intermédiaire de SPIN
- Traduction basée sur les transitions similaire à celle de Couvreur (FM'99)

Contributions sur la traduction (2/3)

Nouvelles règles de simplifications de formules.
Exemple :

motif	conditions à tester	signification	simplification
$\varphi \,\mathsf{U}\, \psi$	$\mathscr{L}(A_{\varphi \,\mathsf{U}\, \psi} \otimes A_{\neg\psi}) = \emptyset$	$\varphi \,\mathsf{U}\, \psi \implies \psi$	ψ
	$\mathscr{L}(A_{\neg\varphi} \otimes A_{\neg\psi}) = \emptyset$	$\neg\varphi \implies \psi$	$\mathsf{F}\, \psi$
$\vdots$	$\vdots$	$\vdots$	$\vdots$

Meilleure simplification que les règles basées sur l'implication de formules existantes.

Contributions sur la traduction (2/3)

Nouvelles règles de simplifications de formules.
Exemple :

motif	conditions à tester	signification	simplification
$\varphi \,\mathsf{U}\, \psi$	$\mathscr{L}(A_{\varphi \,\mathsf{U}\, \psi} \otimes A_{\neg\psi}) = \emptyset$	$\varphi \,\mathsf{U}\, \psi \implies \psi$	ψ
	$\mathscr{L}(A_{\neg\varphi} \otimes A_{\neg\psi}) = \emptyset$	$\neg\varphi \implies \psi$	$\mathsf{F}\, \psi$
$\vdots$	$\vdots$	$\vdots$	$\vdots$

Meilleure simplification que les règles basées sur l'implication de formules existantes.

	formules	
règles de réduction	aléatoires	prédéfinies
classiques	61.59%	7.63%
celles-ci	63.82%	10.18%

Contributions sur la traduction (3/3)

Combinaison avec d'autres optimisations existantes, notamment :

- amélioration du déterminisme quand cela est possible
- simplifications de l'automate produit par simulations

$\Rightarrow$ L'expérience montre la traduction compétitive avec les dernières traductions publiées.

Contributions sur la traduction (3/3)

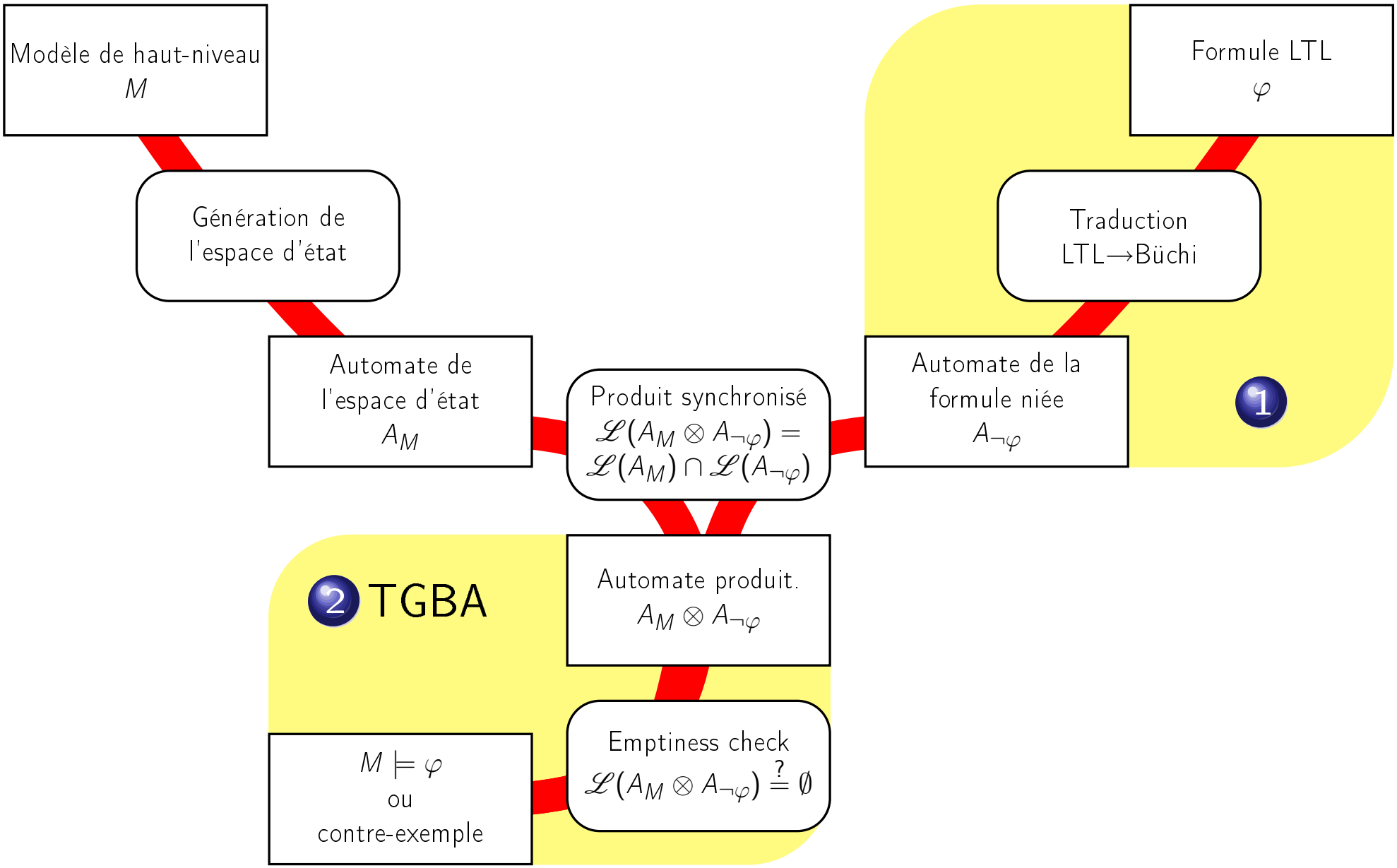
Combinaison avec d'autres optimisations existantes, notamment :

- amélioration du déterminisme quand cela est possible
- simplifications de l'automate produit par simulations

$\Rightarrow$ L'expérience montre la traduction compétitive avec les dernières traductions publiées.

Formules	aléatoires			prédéfinies		
	ét.	tr.	tps	ét.	tr.	tps
Traductions en automates de Büchi dégénéralisés :						
LTL2NBA	499	991	5.7	543	2130	537.0
Spot	505	1023	6.3	527	1639	23.4
Traductions en automates de Büchi généralisés :						
LTL2BA	507	1014	4.3	496	1764	4.2
Spot	479	923	3.8	413	1267	13.8

Approche automate du *model checking*

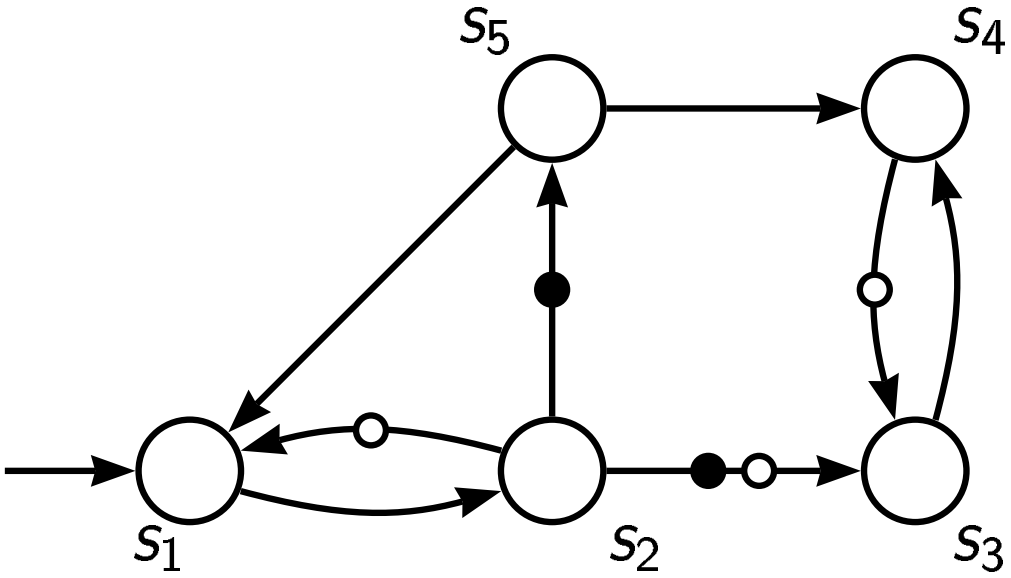


GBA basé sur les transitions

Un TGBA possède :

- un ensemble d'états (dont un état initial),
- un ensemble de transitions entre ces états,
- un ensemble de conditions d'acceptation, ici : $\{\bullet, \circ\}$.

Une exécution infinie est acceptante ssi elle visite infiniment souvent chaque condition d'acceptation.

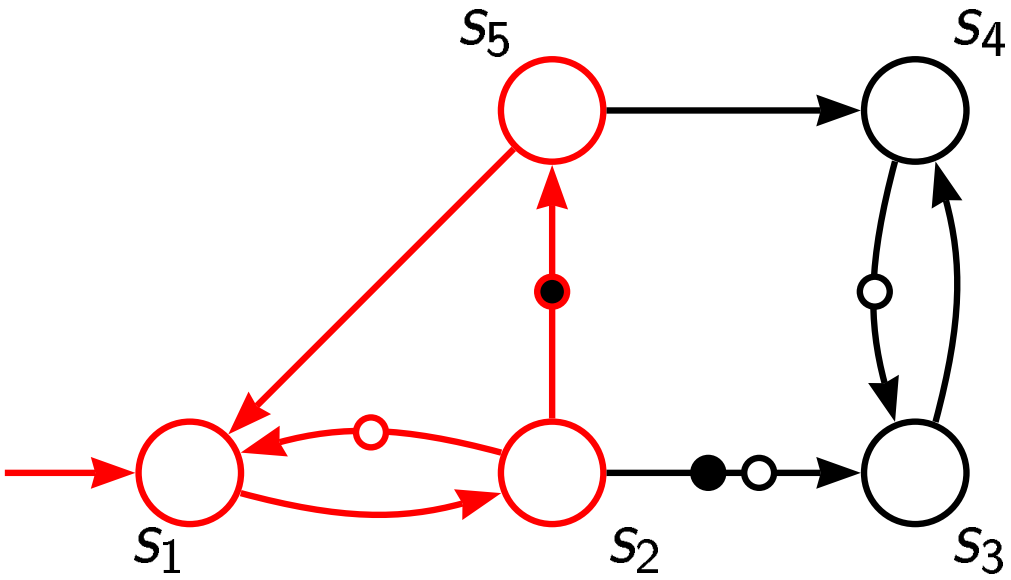


GBA basé sur les transitions

Un TGBA possède :

- un ensemble d'états (dont un état initial),
- un ensemble de transitions entre ces états,
- un ensemble de conditions d'acceptation, ici : $\{\bullet, \circ\}$.

Une exécution infinie est acceptante ssi elle visite infiniment souvent chaque condition d'acceptation.



Algorithmes d'*emptiness checks* : principe

Avec DFS imbriqués (NDFS) :

- Si une seule condition d'acceptation :
  - Un DFS pour chercher une condition d'acceptation
    + un DFS imbriqué pour chercher un circuit.

Algorithmes d'*emptiness checks* : principe

Avec DFS imbriqués (NDFS) :

- Si une seule condition d'acceptation :
  - Un DFS pour chercher une condition d'acceptation
    + un DFS imbriqué pour chercher un circuit.
- Si plus de conditions d'acceptations :
  - Conversion en automate à 1 condition (dégénéralisation)
    pour utiliser la technique précédente,

  ou
  - Un DFS pour chercher des conditions d'acceptations
    + plusieurs DFS imbriqués [Tauriainen '03].

Algorithmes d'*emptiness checks* : principe

Avec DFS imbriqués (NDFS) :

- Si une seule condition d'acceptation :
  - Un DFS pour chercher une condition d'acceptation
    + un DFS imbriqué pour chercher un circuit.
- Si plus de conditions d'acceptations :
  - Conversion en automate à 1 condition (dégénéralisation)
    pour utiliser la technique précédente,

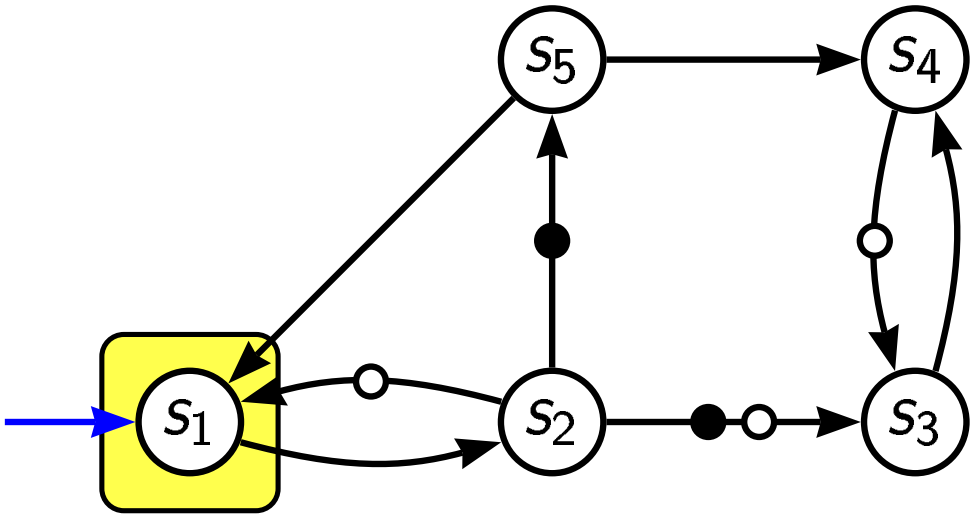
  ou

  - Un DFS pour chercher des conditions d'acceptations
    + plusieurs DFS imbriqués [Taurainen '03].

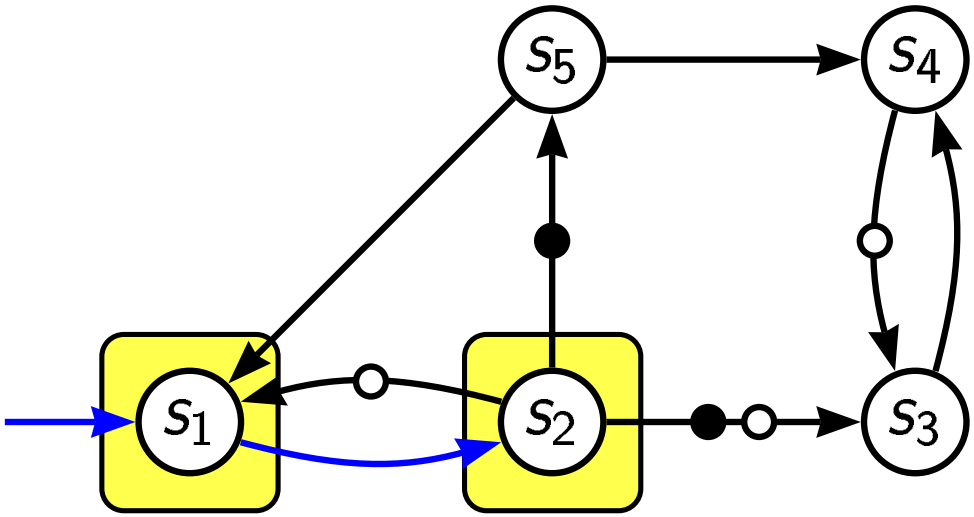
Par recherche de composantes fortement connexes (CFC) :

- On cherche une CFC possédant une transition pour chaque condition d'acceptation.

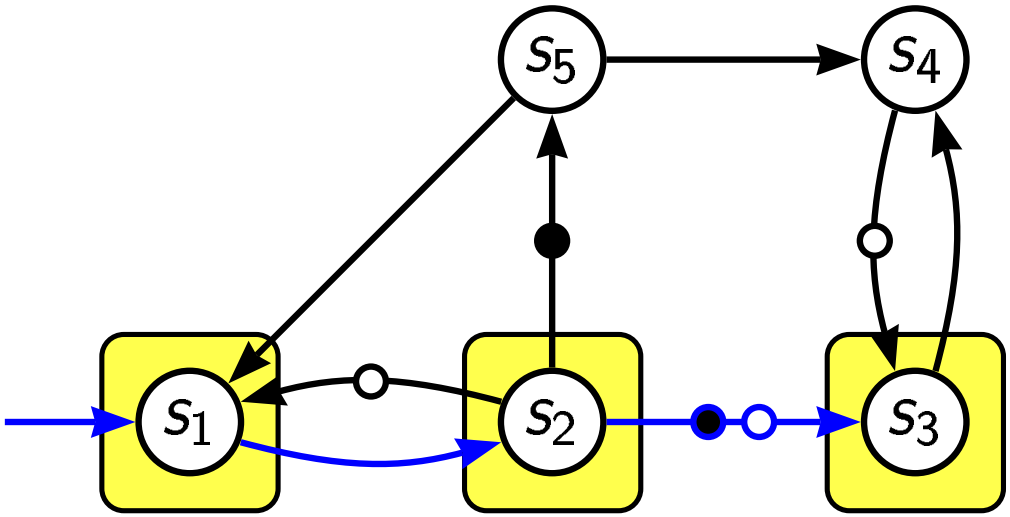
Recherche de CFC : algorithme de Couvreur



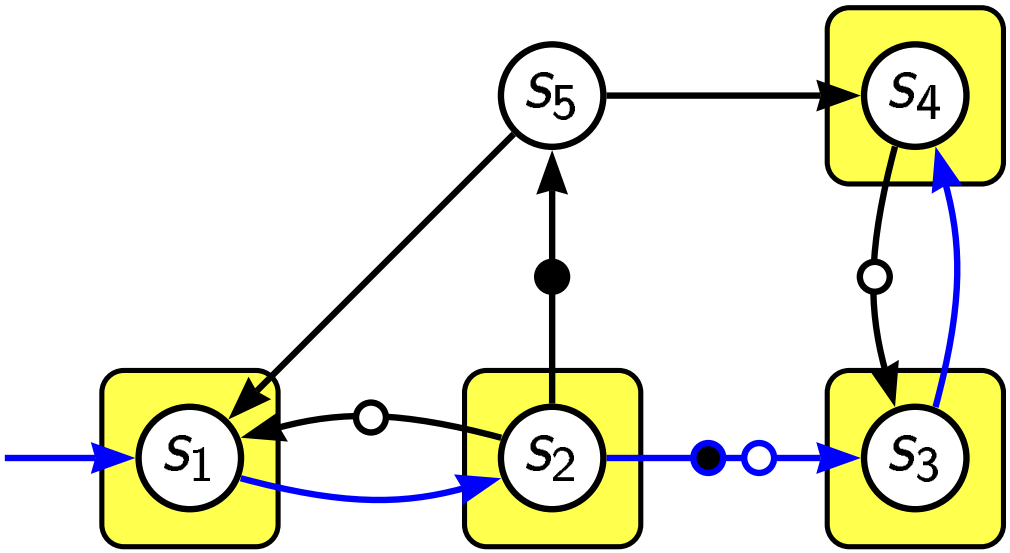
Recherche de CFC : algorithme de Couvreur



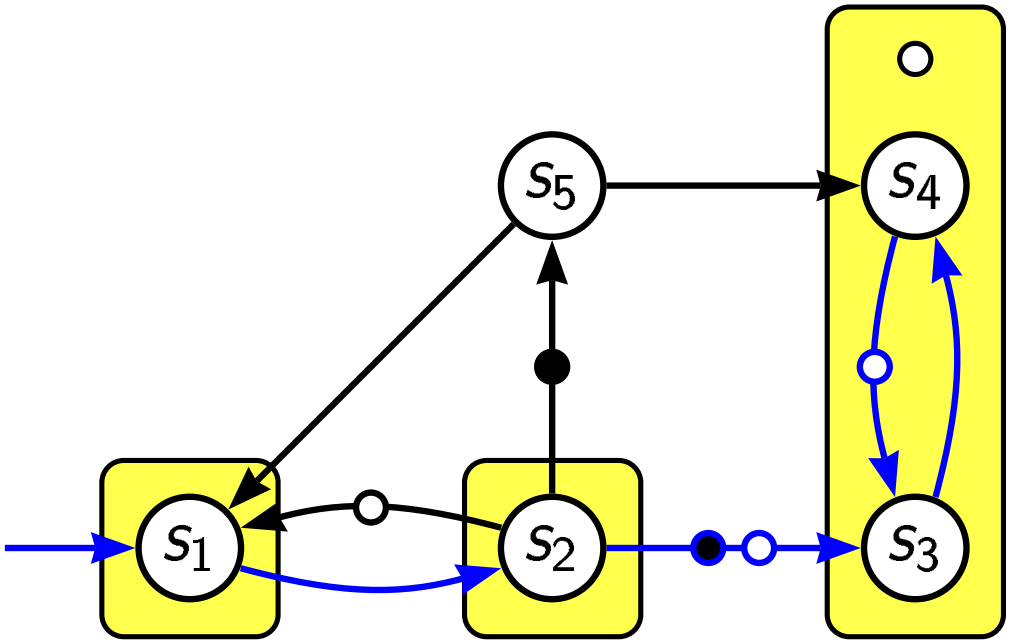
Recherche de CFC : algorithme de Couvreur



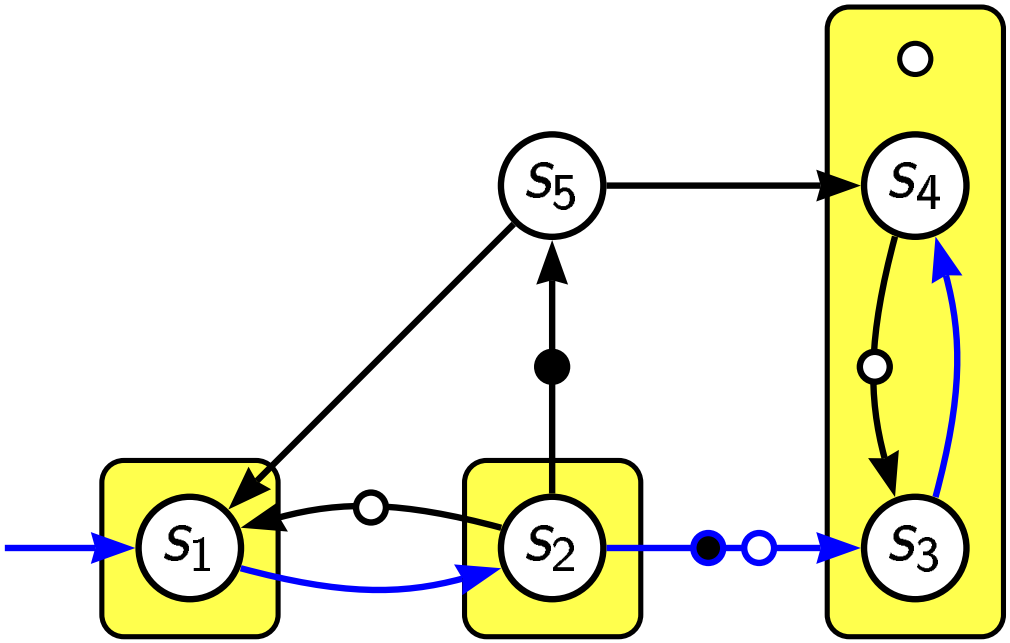
Recherche de CFC : algorithme de Couvreur



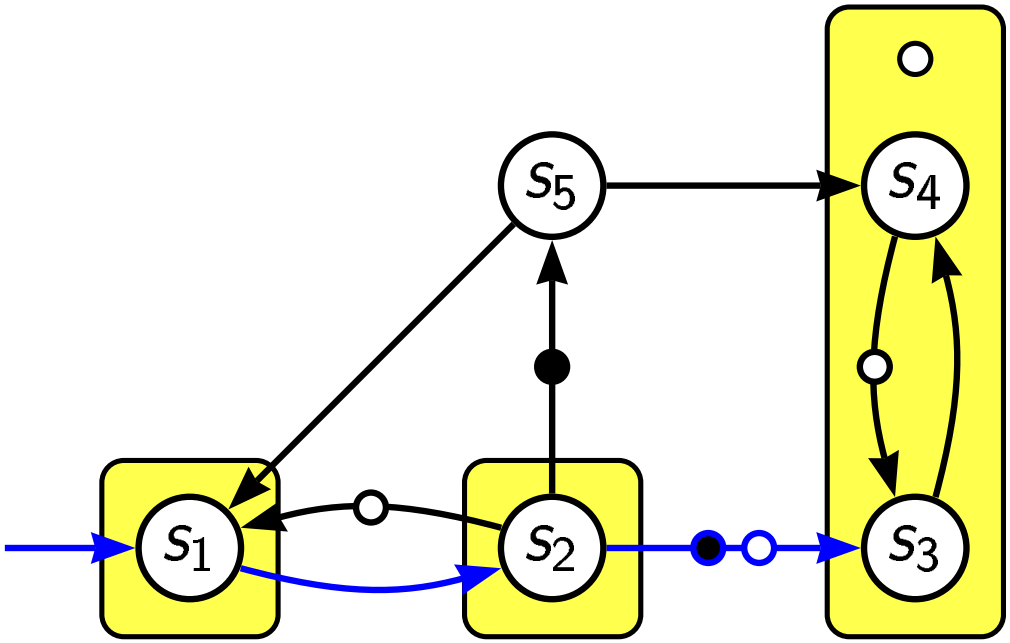
Recherche de CFC : algorithme de Couvreur



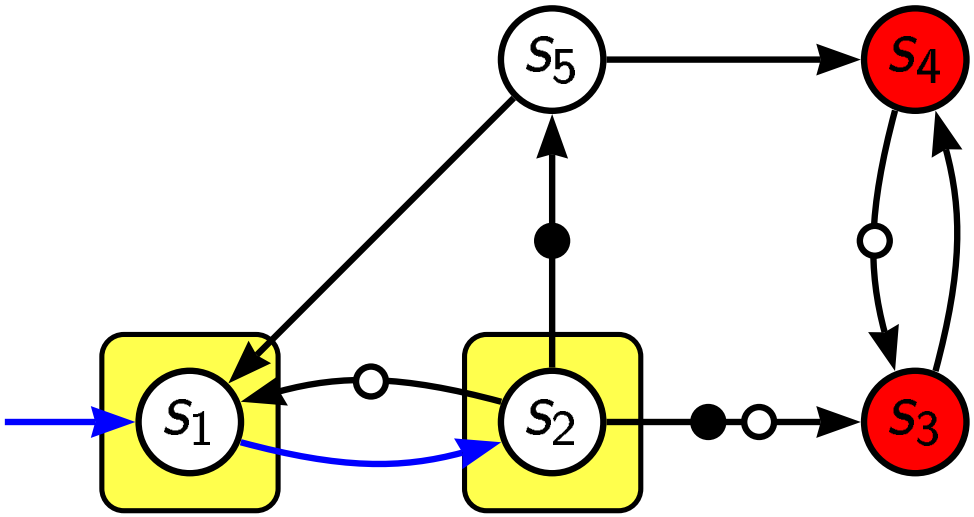
Recherche de CFC : algorithme de Couvreur



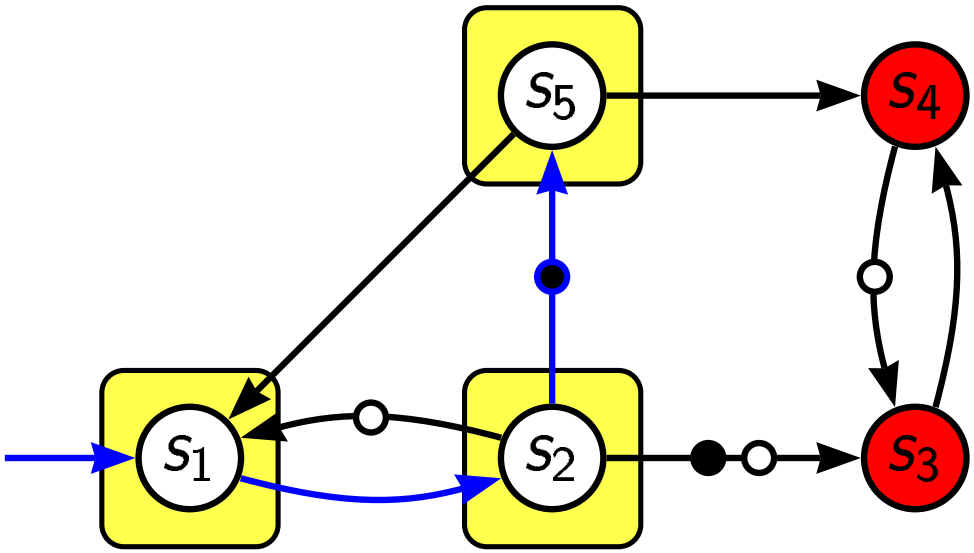
Recherche de CFC : algorithme de Couvreur



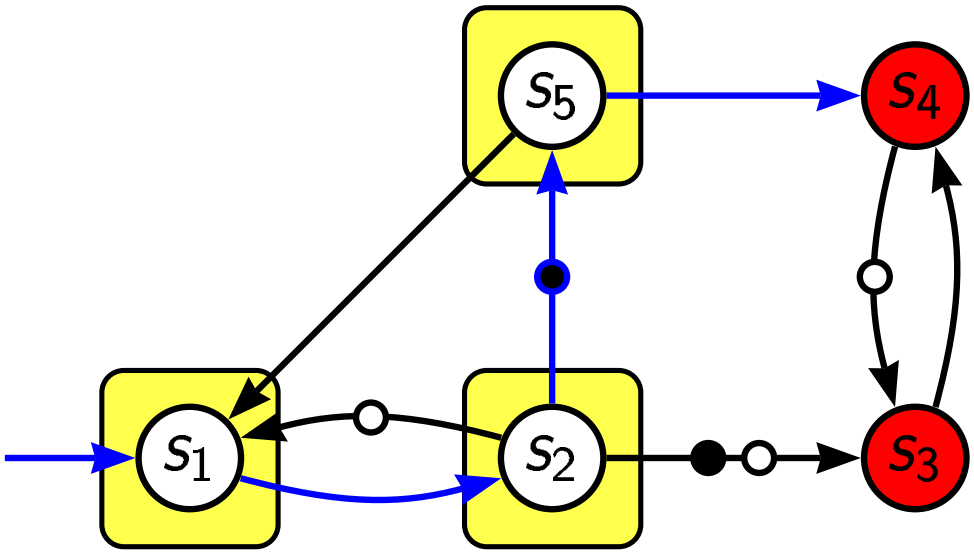
Recherche de CFC : algorithme de Couvreur



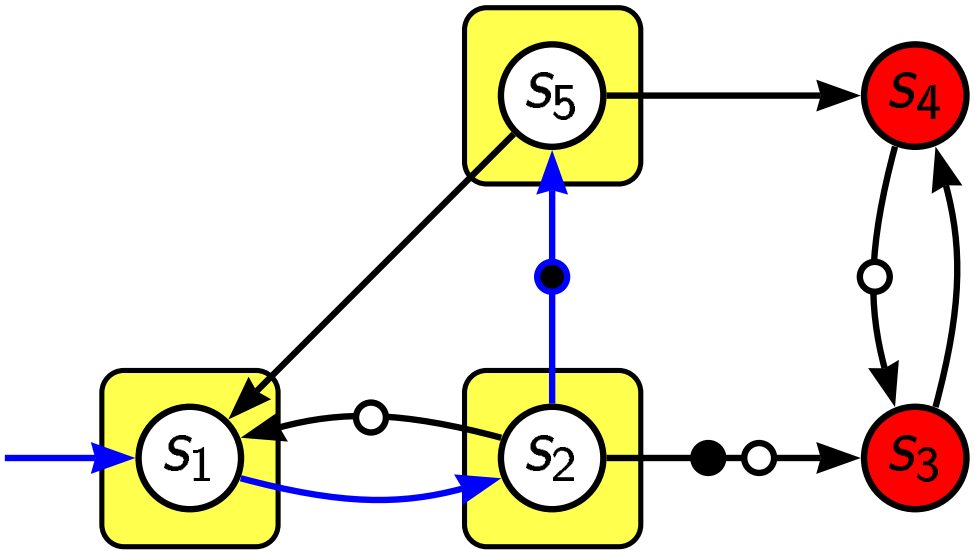
Recherche de CFC : algorithme de Couvreur



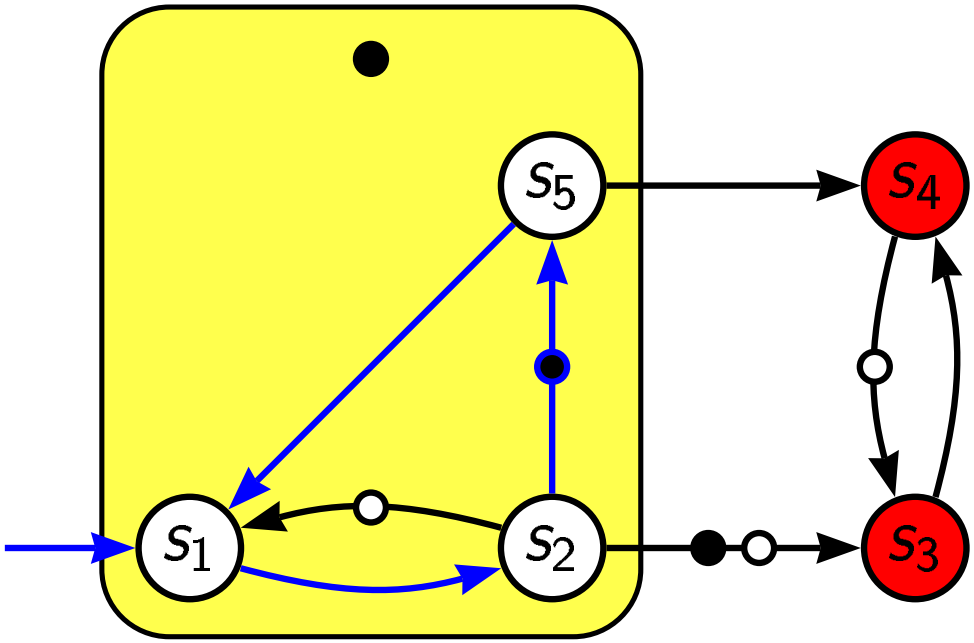
Recherche de CFC : algorithme de Couvreur



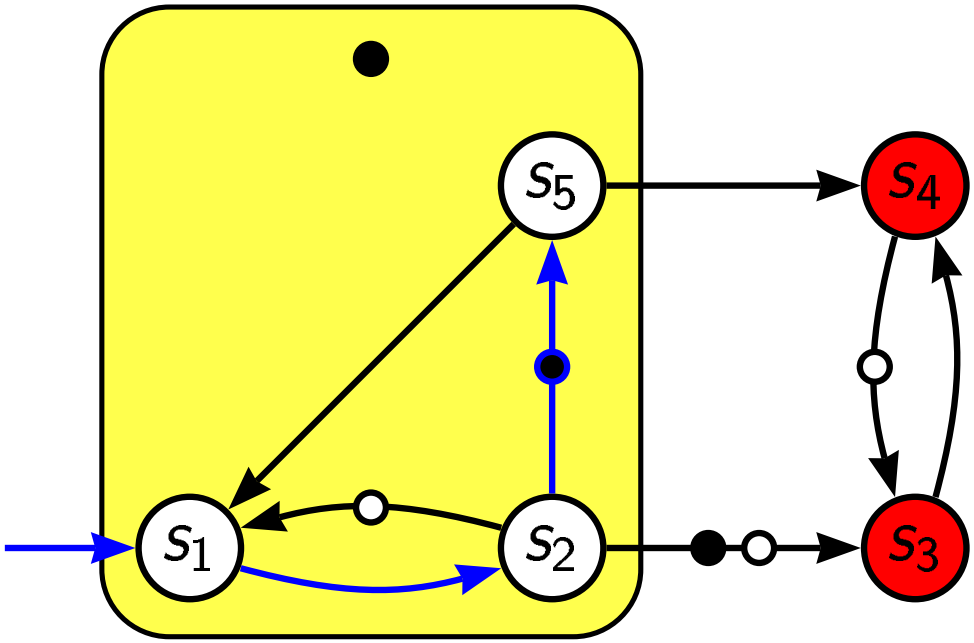
Recherche de CFC : algorithme de Couvreur



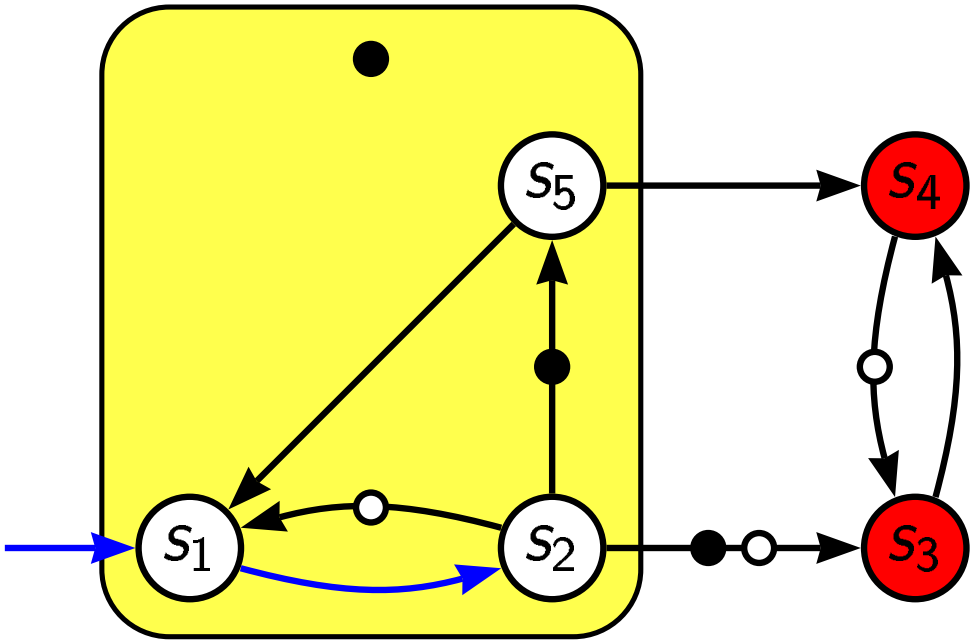
Recherche de CFC : algorithme de Couvreur



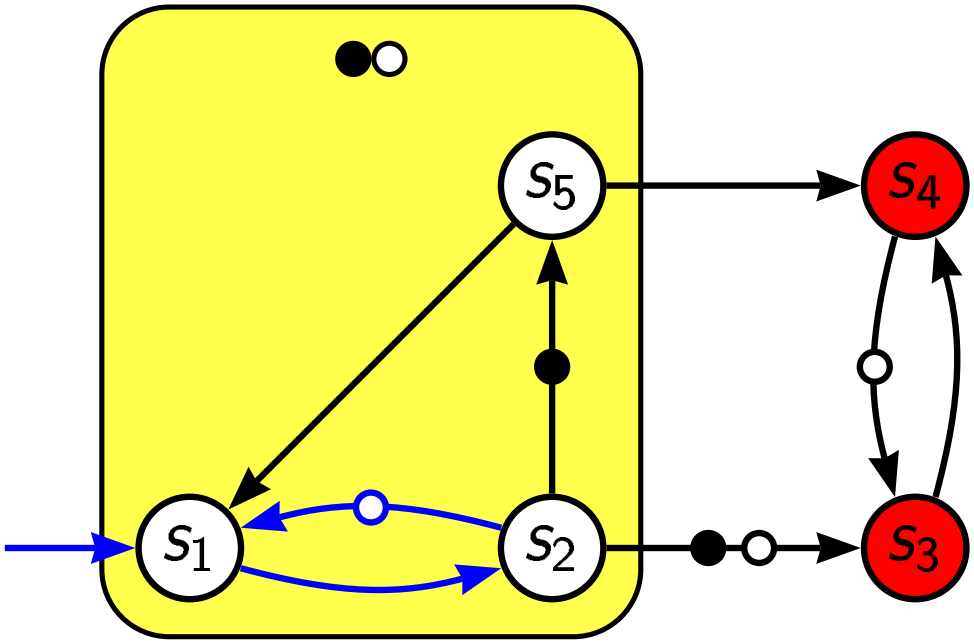
Recherche de CFC : algorithme de Couvreur



Recherche de CFC : algorithme de Couvreur



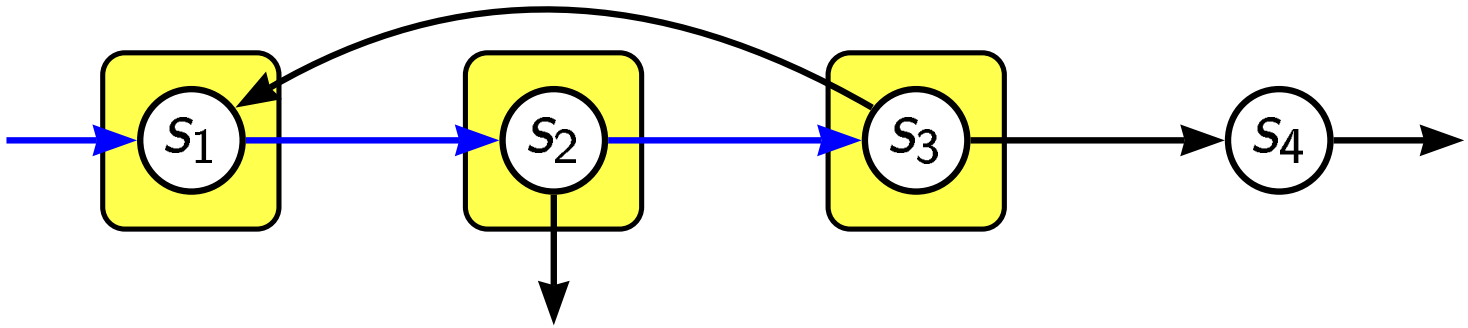
Recherche de CFC : algorithme de Couvreur



Trouvé !

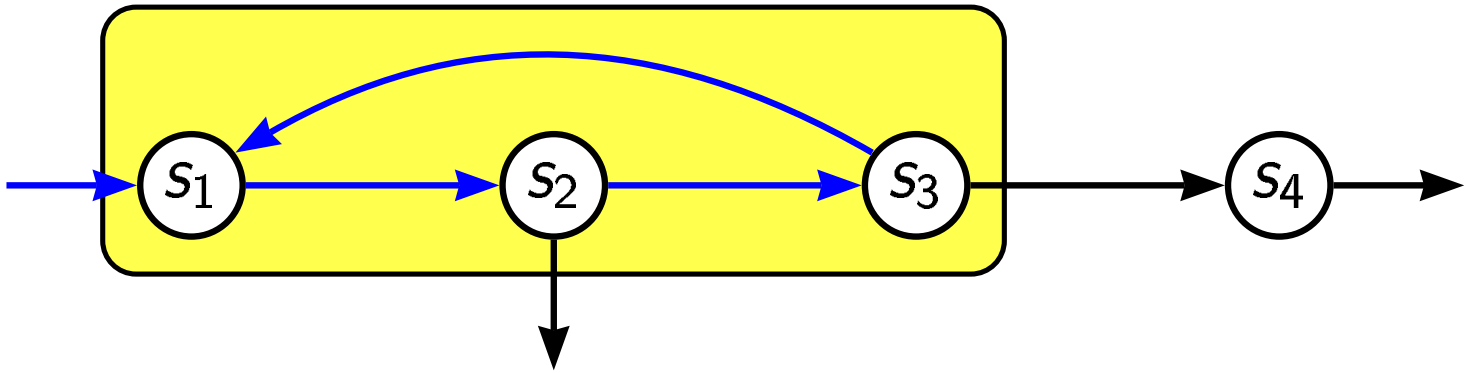
Deux heuristiques pour les CFC

- H1 : visiter d'abord les transitions amenant à des états connus



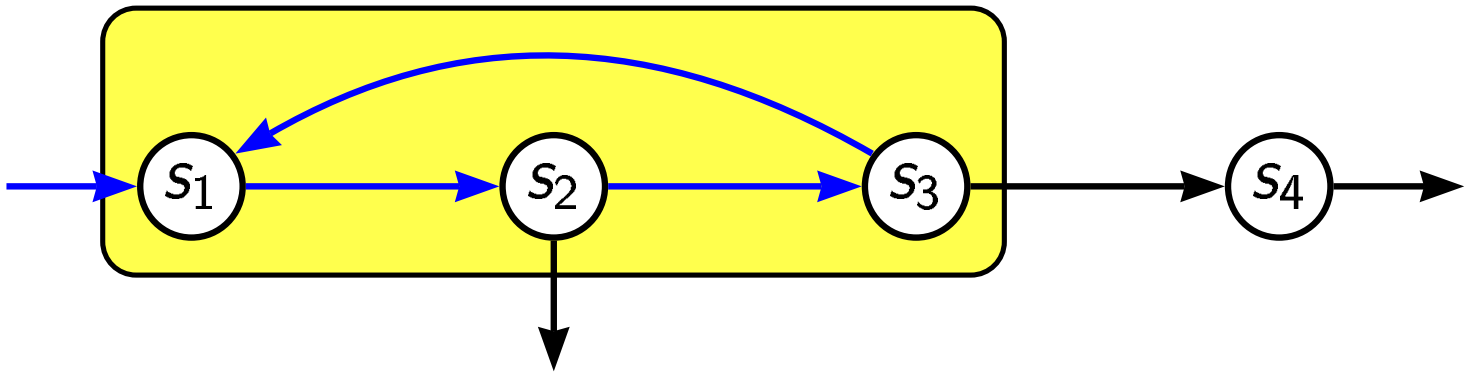
Deux heuristiques pour les CFC

- H1 : visiter d'abord les transitions amenant à des états connus



Deux heuristiques pour les CFC

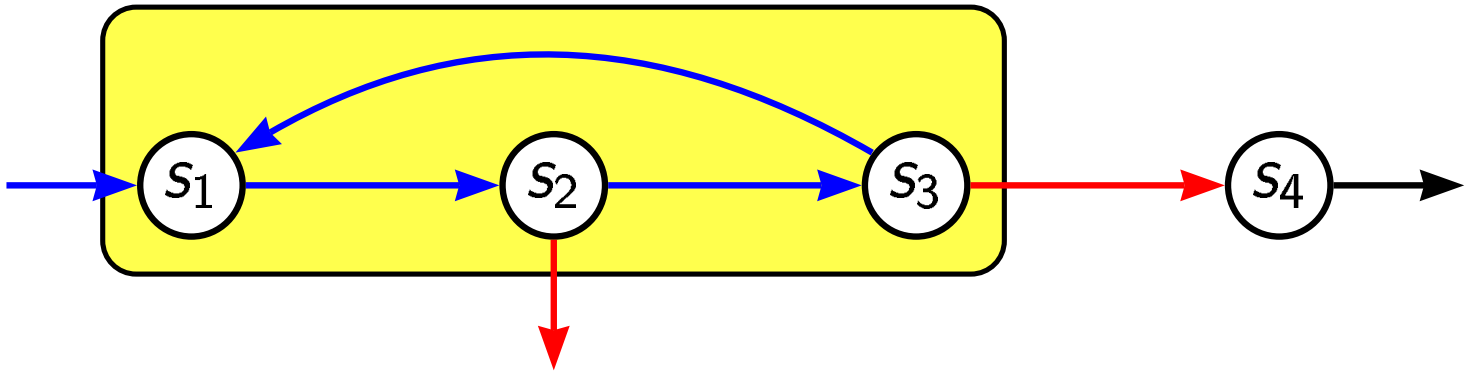
- H1 : visiter d'abord les transitions amenant à des états connus



- H2 : H1 + considérer le DFS en termes de CFC pour le choix des successeurs.

Deux heuristiques pour les CFC

- H1 : visiter d'abord les transitions amenant à des états connus



- H2 : H1 + considérer le DFS en termes de CFC pour le choix des successeurs.

Classification des *emptiness checks*

	dégénéralisé (**une** condition d'acceptation)		généralisé (*m* conditions d'acceptation)	
NDFS	DFS vers transitions acc. + NDFS pour trouver circuit		DFS vers transitions acc. + **plusieurs** NDFS	
	Courcoubetis et al.	'90	Tauriainen	'03
	Godefroid & Holzmann	'93	Couvreur et al.	'05
	Holzmann et al.	'96	Tauriainen	'05
	Gastin et al.	'04		
	Schwoon & Esparza	'04		
CFC	CFC calculées à la volée		CFC calculées à la volée	
	Geldenhuys & Valmari	'04	Couvreur	'99
			Geldenhuys & Valmari	'05
			Couvreur et al.	'05

Classification des *emptiness checks*

	dégénéralisé (**une** condition d'acceptation)	généralisé (m conditions d'acceptation)
NDFS	DFS vers transitions acc. + NDFS pour trouver circuit • 2 bits par état • contre-exemples immédiats • dégénéralisation requise (taille $\times m$) • lent	DFS vers transitions acc. + **plusieurs** NDFS • $\log_2(m+1)$ bits par état • états visités plusieurs fois • lent • contre-ex. à calculer
CFC	CFC calculées à la volée • sans intérêt	CFC calculées à la volée • plus rapide • états visités une seule fois • indépendant de m • contre-ex. à calculer • un entier par état

Benchmarks des *emptiness checks*

dégénéralisés	Algorithme	Formules prédéfinies: cond. acc. LTL %ét.	%tr.	tps	cond. acc. supp. %ét.	%tr.	tps	
	Cou99	7.4	5.1	4.0	16.3	10.6	10.4	CFC
×	Cou99 shy-	6.5	6.8	7.7	15.2	15.7	19.6	CFC
×	Cou99 shy	6.3	6.5	7.0	14.5	15.0	18.0	CFC
	GV04	7.5	5.1	4.0	25.9	16.5	16.1	CFC
	CVWY90	7.7	7.9	5.2	53.6	65.0	49.3	NDFS
	SE05	7.6	6.8	3.9	50.9	34.7	28.1	NDFS
	Tau03	9.5	17.4	8.1	53.9	265.5	36.2	NDFS
×	Tau03 opt	7.4	8.1	3.8	16.4	31.8	11.3	NDFS

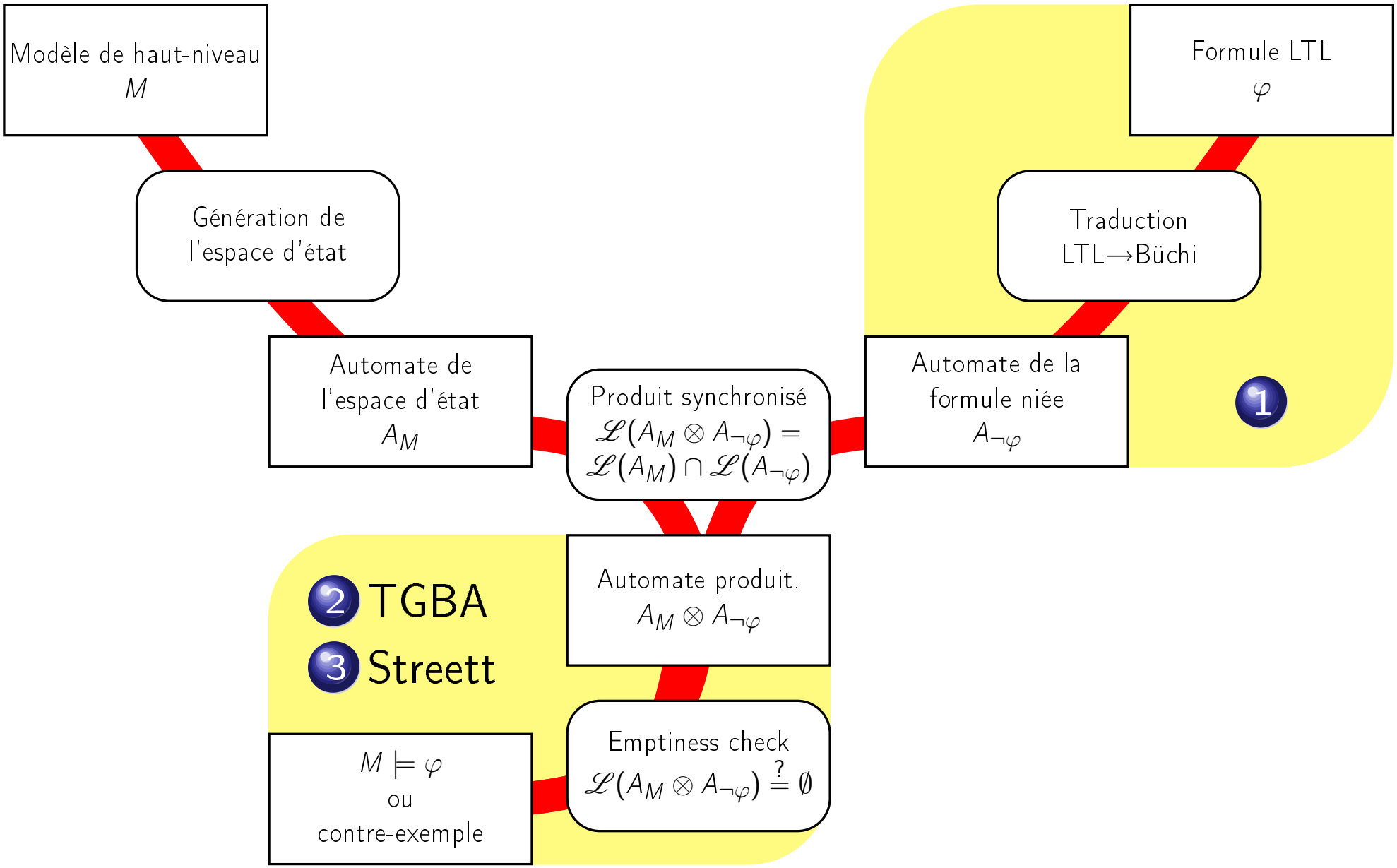
Emptiness checks : contributions

SPIN'05 — ACSD'07

- Benchmarks importants des algorithmes existants.
- Pour l'algorithme de Couvreur :
  - Heuristiques pour favoriser les fusions de CFC.
  - Algorithme pour le calcul de contre-exemple.
- Pour les NDFS :
  - Utilisation de *poids* pour détecter des circuits acceptants plus rapidement.
  - Amélioration de l'algorithme de Tauriainen.
  - Algorithme pour le calcul de contre-exemple dans le cas généralisé.

- Variante de l'algorithme de Couvreur pour « automates ensemblistes » avec tests d'inclusion entre les états.

Approche automate du *model checking*



Équité faible et forte

Contraintes exprimable en LTL

- Équité faible : $\mathsf{F}\,\mathsf{G}\,a \Rightarrow \mathsf{G}\,\mathsf{F}\,b$
  si a se produit **continûment**, b se produira infiniment souvent

- Équité forte : $\mathsf{G}\,\mathsf{F}\,a \Rightarrow \mathsf{G}\,\mathsf{F}\,b$
  si a se produit **infiniment souvent**, b se produira infiniment souvent

Équité faible et forte

Contraintes exprimable en LTL

- Équité faible : $\bigwedge_{i=1}^{m} \mathsf{F}\,\mathsf{G}\,a_i \Rightarrow \mathsf{G}\,\mathsf{F}\,b_i$
  si a_i se produit **continûment**, b_i se produira infiniment souvent

- Équité forte : $\bigwedge_{i=1}^{m} \mathsf{G}\,\mathsf{F}\,a_i \Rightarrow \mathsf{G}\,\mathsf{F}\,b_i$
  si a_i se produit **infiniment souvent**, b_i se produira infiniment souvent

Équité faible et forte

Contraintes exprimable en LTL

- Équité faible : $\bigwedge_{i=1}^{m} \mathsf{F}\,\mathsf{G}\,a_i \Rightarrow \mathsf{G}\,\mathsf{F}\,b_i$
  si a_i se produit **continûment**, b_i se produira infiniment souvent
  - Automate de Büchi généralisé étiqueté sur les états avec m états, m conditions d'acceptations, non-déterministe.
  - Automate de Büchi généralisé étiqueté sur les transitions avec 1 état, m conditions d'acceptations, déterministe.
- Équité forte : $\bigwedge_{i=1}^{m} \mathsf{G}\,\mathsf{F}\,a_i \Rightarrow \mathsf{G}\,\mathsf{F}\,b_i$
  si a_i se produit **infiniment souvent**, b_i se produira infiniment souvent

Équité faible et forte

Contraintes exprimable en LTL

- Équité faible : $\bigwedge_{i=1}^{m} \mathsf{F}\,\mathsf{G}\,a_i \Rightarrow \mathsf{G}\,\mathsf{F}\,b_i$
  si a_i se produit **continûment**, b_i se produira infiniment souvent
  - Automate de Büchi généralisé étiqueté sur les états avec m états, m conditions d'acceptations, non-déterministe.
  - Automate de Büchi généralisé étiqueté sur les transitions avec 1 état, m conditions d'acceptations, déterministe.
- Équité forte : $\bigwedge_{i=1}^{m} \mathsf{G}\,\mathsf{F}\,a_i \Rightarrow \mathsf{G}\,\mathsf{F}\,b_i$
  si a_i se produit **infiniment souvent**, b_i se produira infiniment souvent
  - Automate de Büchi généralisé étiqueté sur les transitions avec $3^m + 1$ états, $2m$ conditions d'acceptation, non-déterministe.

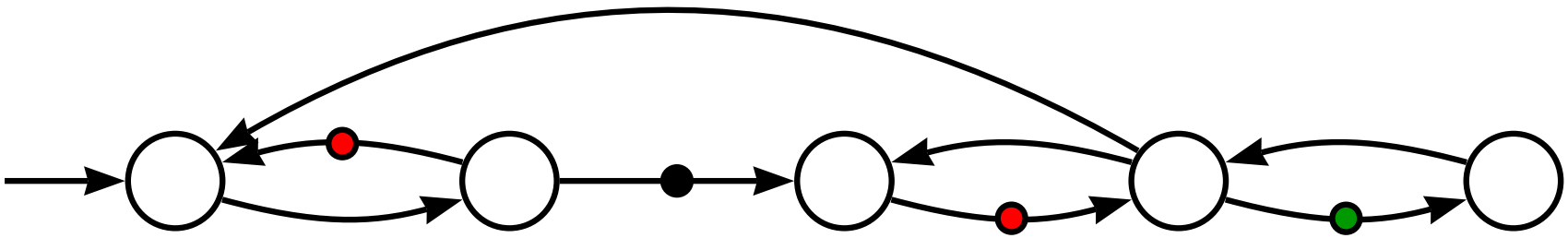
Équité faible et forte

Contraintes exprimable en LTL

- Équité faible : $\bigwedge_{i=1}^{m} \mathsf{F}\,\mathsf{G}\,a_i \Rightarrow \mathsf{G}\,\mathsf{F}\,b_i$
  si a_i se produit **continûment**, b_i se produira infiniment souvent
  - Automate de Büchi généralisé étiqueté sur les états avec m états, m conditions d'acceptations, non-déterministe.
  - Automate de Büchi généralisé étiqueté sur les transitions avec 1 état, m conditions d'acceptations, déterministe.
- Équité forte : $\bigwedge_{i=1}^{m} \mathsf{G}\,\mathsf{F}\,a_i \Rightarrow \mathsf{G}\,\mathsf{F}\,b_i$
  si a_i se produit **infiniment souvent**, b_i se produira infiniment souvent
  - Automate de Büchi généralisé étiqueté sur les transitions avec $3^m + 1$ états, $2m$ conditions d'acceptation, non-déterministe.
  - Automate de Streett étiqueté sur les transitions avec 1 état, m paires de conditions d'acceptation, déterministe.

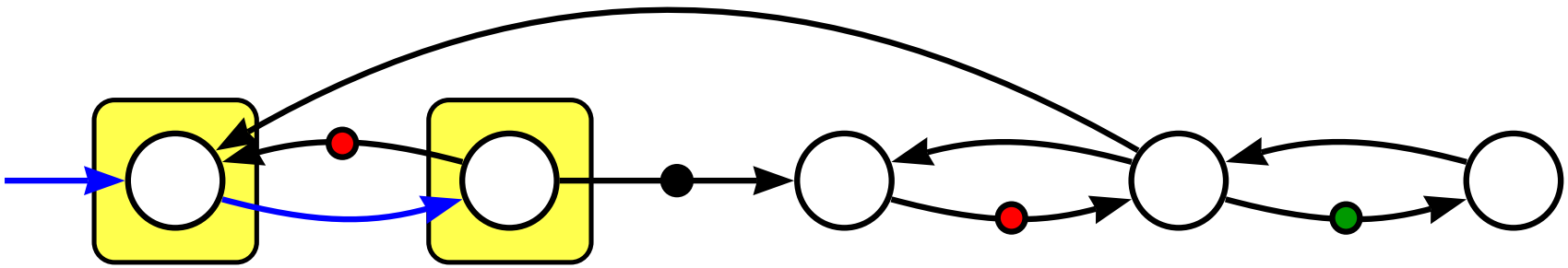
Emptiness check pour automates de Streett

- Pour Büchi, on cherche des CFC dont les conditions d'acceptation vérifient $\bullet \wedge \circ \wedge \bullet \wedge \bullet$.
- Pour Streett, elle doivent vérifier $\bullet \Rightarrow \circ \wedge \bullet \Rightarrow \bullet$.



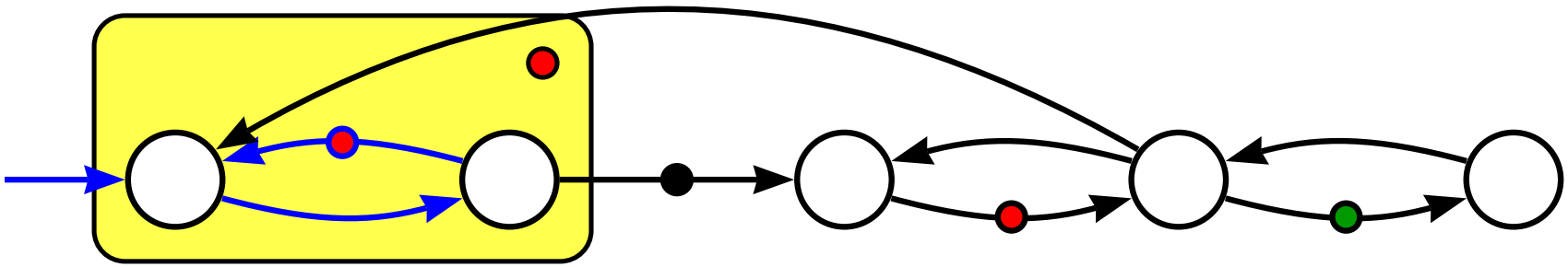
Emptiness check pour automates de Streett

- Pour Büchi, on cherche des CFC dont les conditions d'acceptation vérifient $\bullet \wedge \circ \wedge \bullet \wedge \bullet$.
- Pour Streett, elle doivent vérifier $\bullet \Rightarrow \circ \wedge \bullet \Rightarrow \bullet$.



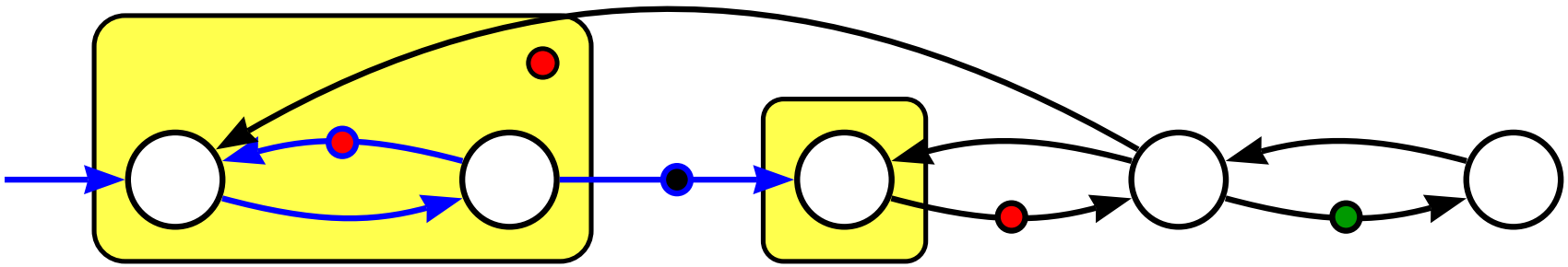
Emptiness check pour automates de Streett

- Pour Büchi, on cherche des CFC dont les conditions d'acceptation vérifient $\bullet \wedge \circ \wedge \bullet \wedge \bullet$.
- Pour Streett, elle doivent vérifier $\bullet \Rightarrow \circ \wedge \bullet \Rightarrow \bullet$.



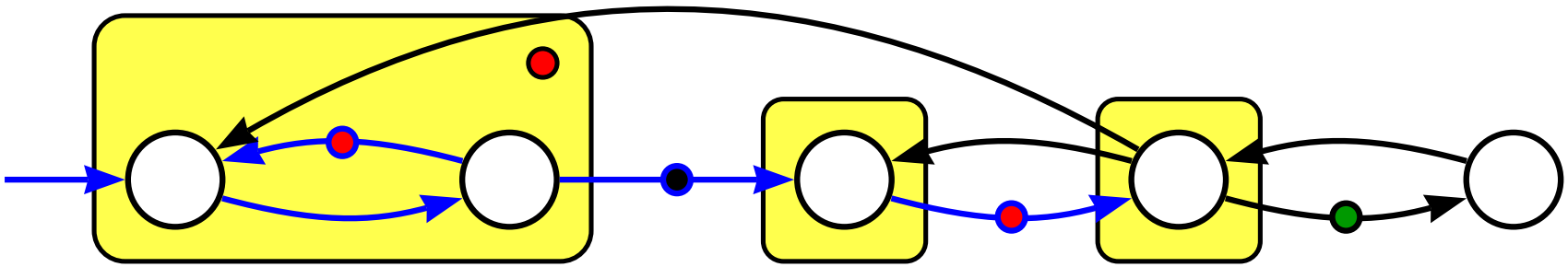
Emptiness check pour automates de Streett

- Pour Büchi, on cherche des CFC dont les conditions d'acceptation vérifient $\bullet \wedge \circ \wedge \bullet \wedge \bullet$.
- Pour Streett, elle doivent vérifier $\bullet \Rightarrow \circ \wedge \bullet \Rightarrow \bullet$.



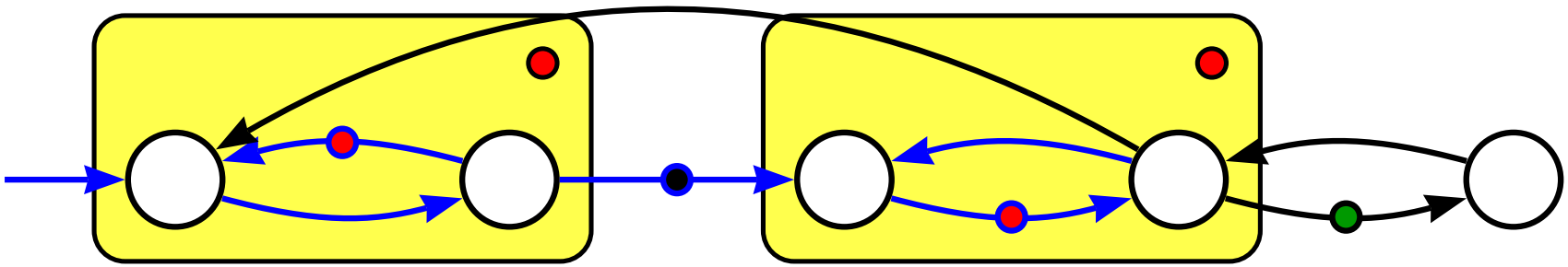
Emptiness check pour automates de Streett

- Pour Büchi, on cherche des CFC dont les conditions d'acceptation vérifient $\bullet \wedge \circ \wedge \bullet \wedge \bullet$.
- Pour Streett, elle doivent vérifier $\bullet \Rightarrow \circ \wedge \bullet \Rightarrow \bullet$.



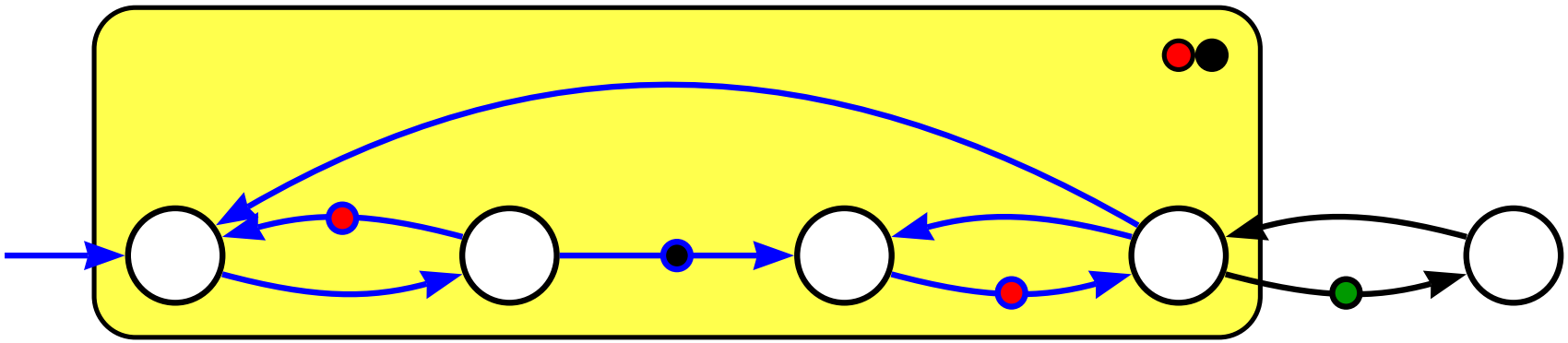
Emptiness check pour automates de Streett

- Pour Büchi, on cherche des CFC dont les conditions d'acceptation vérifient $\bullet \wedge \circ \wedge \bullet \wedge \bullet$.
- Pour Streett, elle doivent vérifier $\bullet \Rightarrow \circ \wedge \bullet \Rightarrow \bullet$.



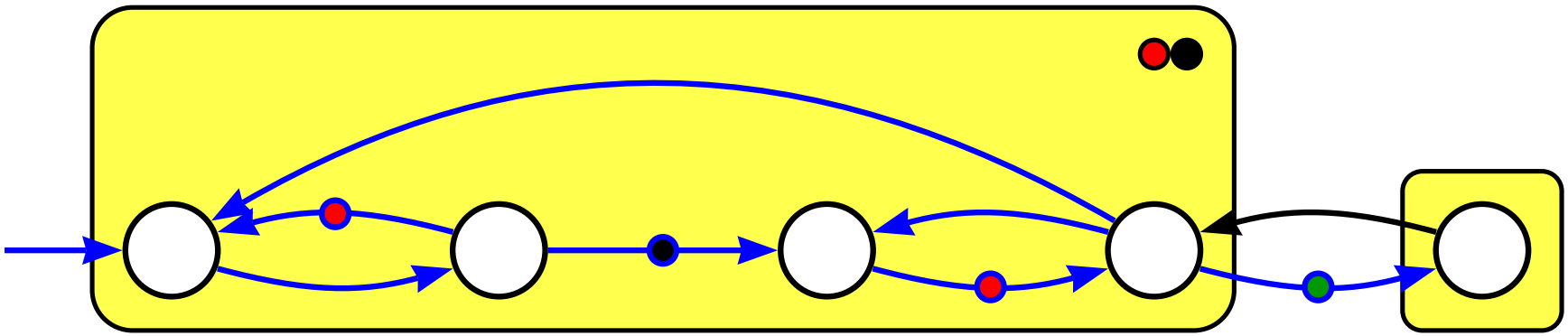
Emptiness check pour automates de Streett

- Pour Büchi, on cherche des CFC dont les conditions d'acceptation vérifient $\bullet \wedge \circ \wedge \bullet \wedge \bullet$.
- Pour Streett, elle doivent vérifier $\bullet \Rightarrow \circ \wedge \bullet \Rightarrow \bullet$.



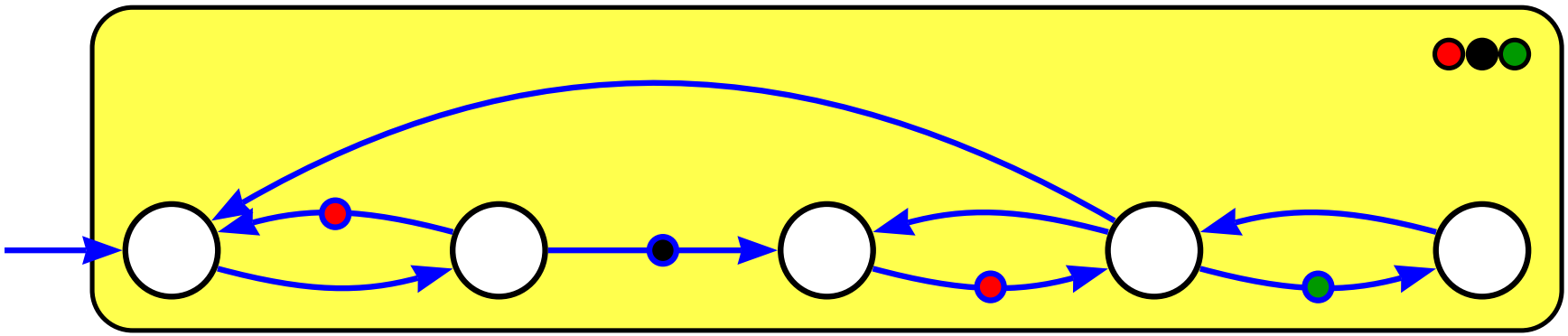
Emptiness check pour automates de Streett

- Pour Büchi, on cherche des CFC dont les conditions d'acceptation vérifient $\bullet \wedge \circ \wedge \bullet \wedge \bullet$.
- Pour Streett, elle doivent vérifier $\bullet \Rightarrow \circ \wedge \bullet \Rightarrow \bullet$.



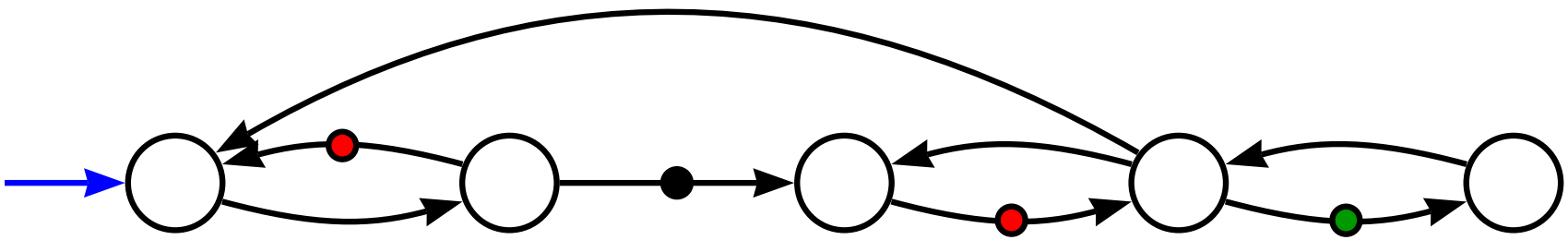
Emptiness check pour automates de Streett

- Pour Büchi, on cherche des CFC dont les conditions d'acceptation vérifient $\bullet \wedge \circ \wedge \bullet \wedge \bullet$.
- Pour Streett, elle doivent vérifier $\bullet \Rightarrow \circ \wedge \bullet \Rightarrow \bullet$.



Emptiness check pour automates de Streett

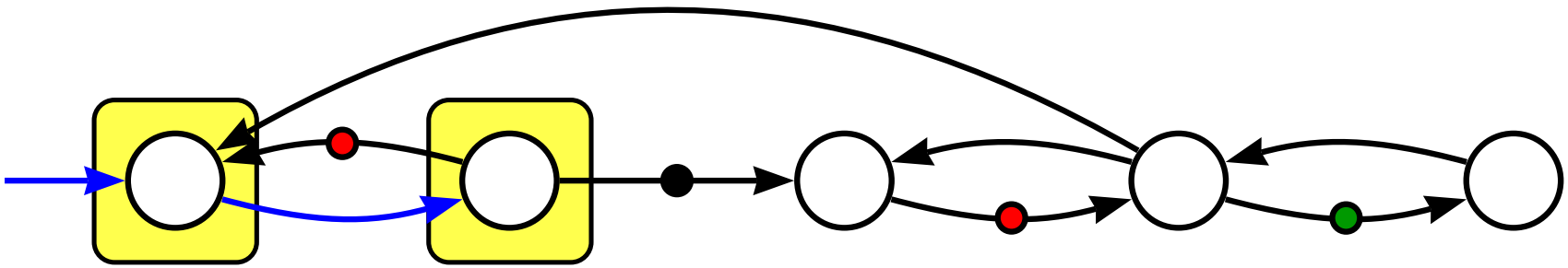
- Pour Büchi, on cherche des CFC dont les conditions d'acceptation vérifient $\bullet \wedge \circ \wedge \bullet \wedge \bullet$.
- Pour Streett, elle doivent vérifier $\bullet \Rightarrow \circ \wedge \bullet \Rightarrow \bullet$.



Comme il n'y a pas de $\circ$ dans cette CFC, on recommence, en considérant $\bullet$ comme une barrière à sens unique, interdisant les fusions de CFC.

Emptiness check pour automates de Streett

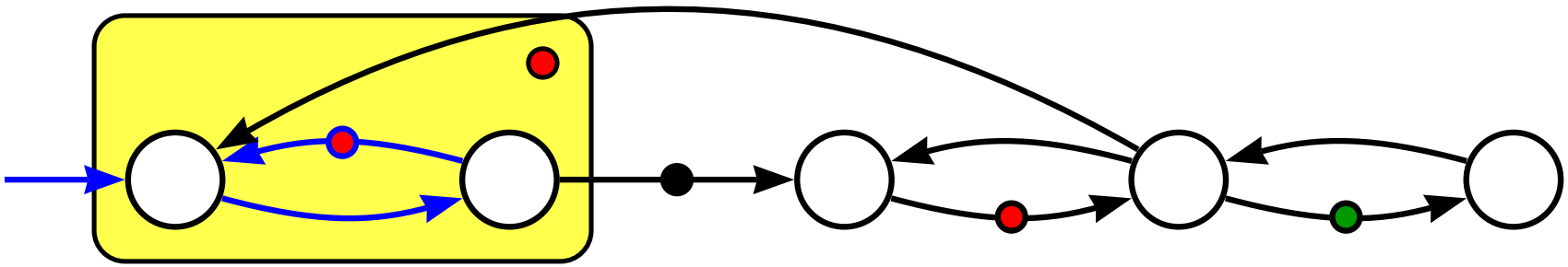
- Pour Büchi, on cherche des CFC dont les conditions d'acceptation vérifient $\bullet \wedge \circ \wedge \bullet \wedge \bullet$.
- Pour Streett, elle doivent vérifier $\bullet \Rightarrow \circ \wedge \bullet \Rightarrow \bullet$.



Comme il n'y a pas de $\circ$ dans cette CFC, on recommence, en considérant $\bullet$ comme une barrière à sens unique, interdisant les fusions de CFC.

Emptiness check pour automates de Streett

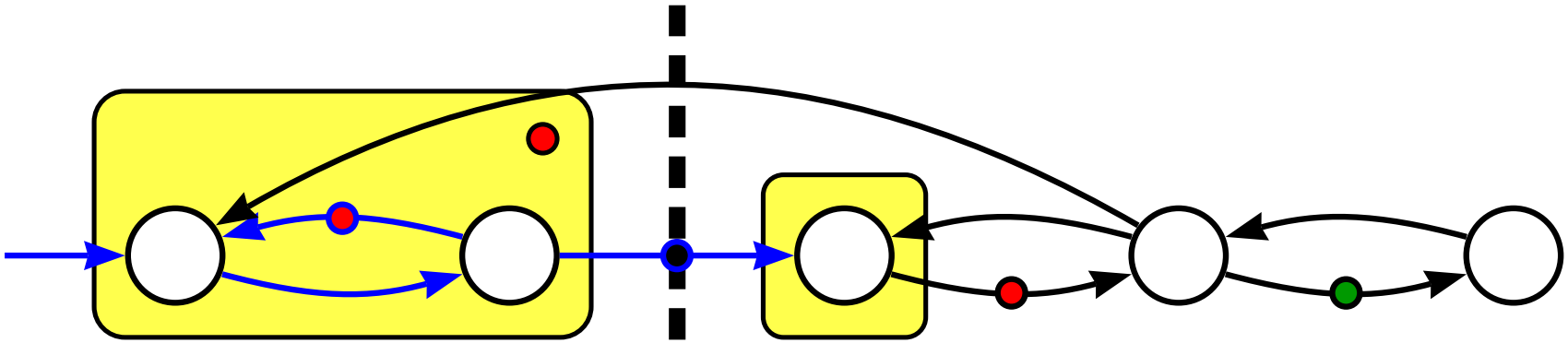
- Pour Büchi, on cherche des CFC dont les conditions d'acceptation vérifient $\bullet \wedge \circ \wedge \bullet \wedge \bullet$.
- Pour Streett, elle doivent vérifier $\bullet \Rightarrow \circ \wedge \bullet \Rightarrow \bullet$.



Comme il n'y a pas de $\circ$ dans cette CFC, on recommence, en considérant $\bullet$ comme une barrière à sens unique, interdisant les fusions de CFC.

Emptiness check pour automates de Streett

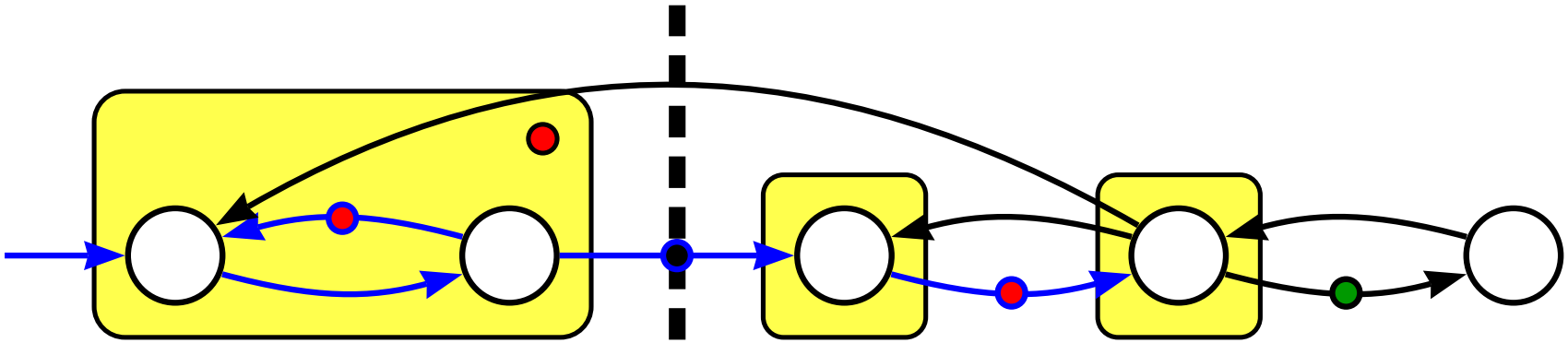
- Pour Büchi, on cherche des CFC dont les conditions d'acceptation vérifient $\bullet \wedge \circ \wedge \bullet \wedge \bullet$.
- Pour Streett, elle doivent vérifier $\bullet \Rightarrow \circ \wedge \bullet \Rightarrow \bullet$.



Comme il n'y a pas de $\circ$ dans cette CFC, on recommence, en considérant $\bullet$ comme une barrière à sens unique, interdisant les fusions de CFC.

Emptiness check pour automates de Streett

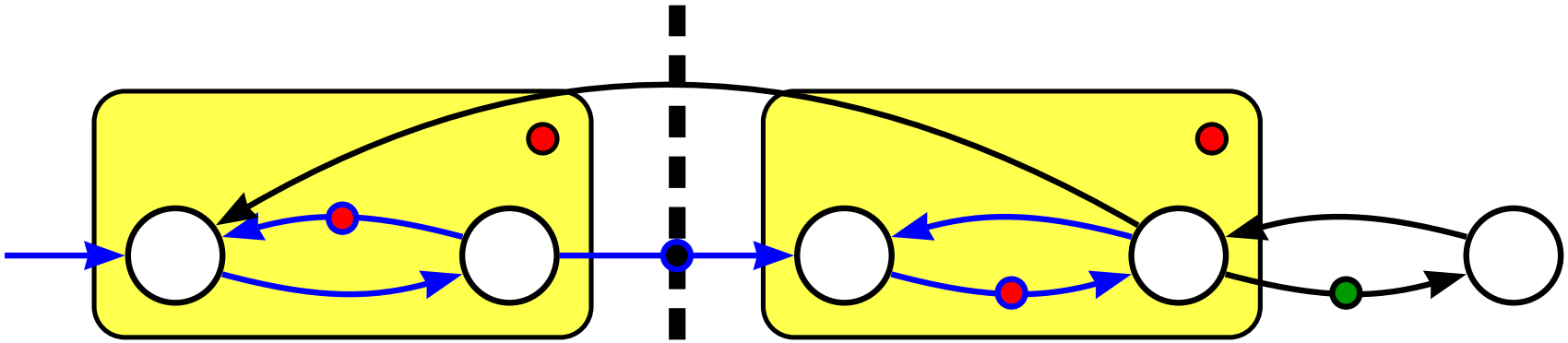
- Pour Büchi, on cherche des CFC dont les conditions d'acceptation vérifient $\bullet \wedge \circ \wedge \bullet \wedge \bullet$.
- Pour Streett, elle doivent vérifier $\bullet \Rightarrow \circ \wedge \bullet \Rightarrow \bullet$.



Comme il n'y a pas de $\circ$ dans cette CFC, on recommence, en considérant $\bullet$ comme une barrière à sens unique, interdisant les fusions de CFC.

Emptiness check pour automates de Streett

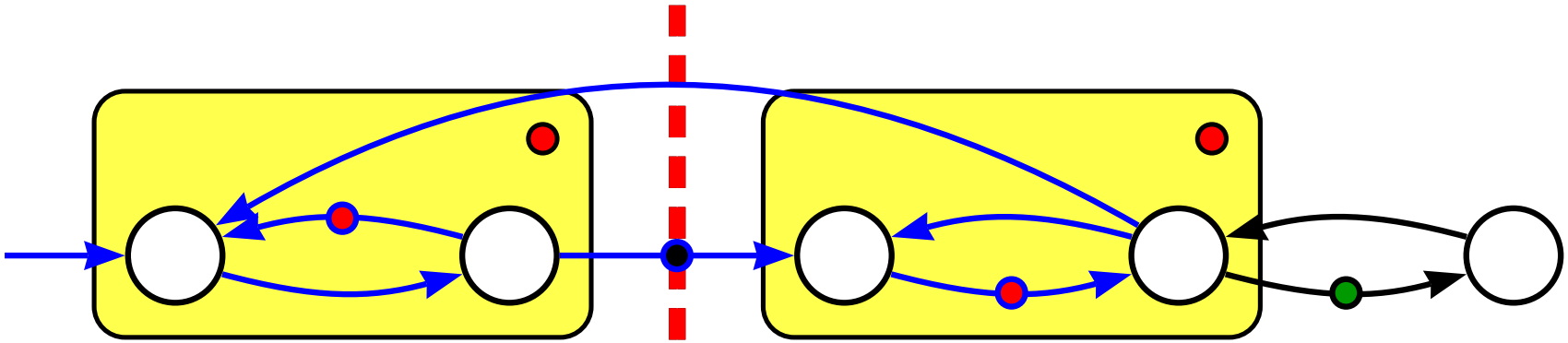
- Pour Büchi, on cherche des CFC dont les conditions d'acceptation vérifient $\bullet \wedge \circ \wedge \bullet \wedge \bullet$.
- Pour Streett, elle doivent vérifier $\bullet \Rightarrow \circ \wedge \bullet \Rightarrow \bullet$.



Comme il n'y a pas de $\circ$ dans cette CFC, on recommence, en considérant $\bullet$ comme une barrière à sens unique, interdisant les fusions de CFC.

Emptiness check pour automates de Streett

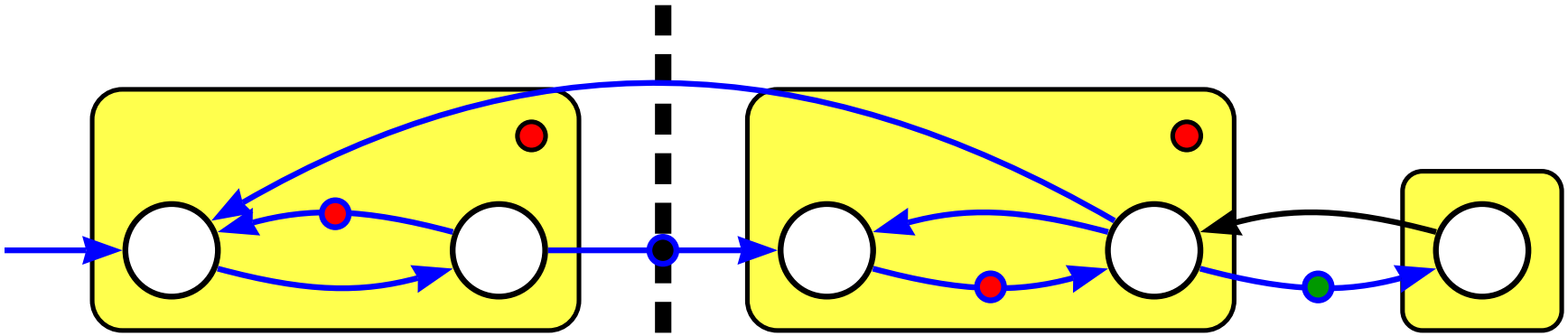
- Pour Büchi, on cherche des CFC dont les conditions d'acceptation vérifient $\bullet \wedge \circ \wedge \bullet \wedge \bullet$.
- Pour Streett, elle doivent vérifier $\bullet \Rightarrow \circ \wedge \bullet \Rightarrow \bullet$.



Comme il n'y a pas de $\circ$ dans cette CFC, on recommence, en considérant $\bullet$ comme une barrière à sens unique, interdisant les fusions de CFC.

Emptiness check pour automates de Streett

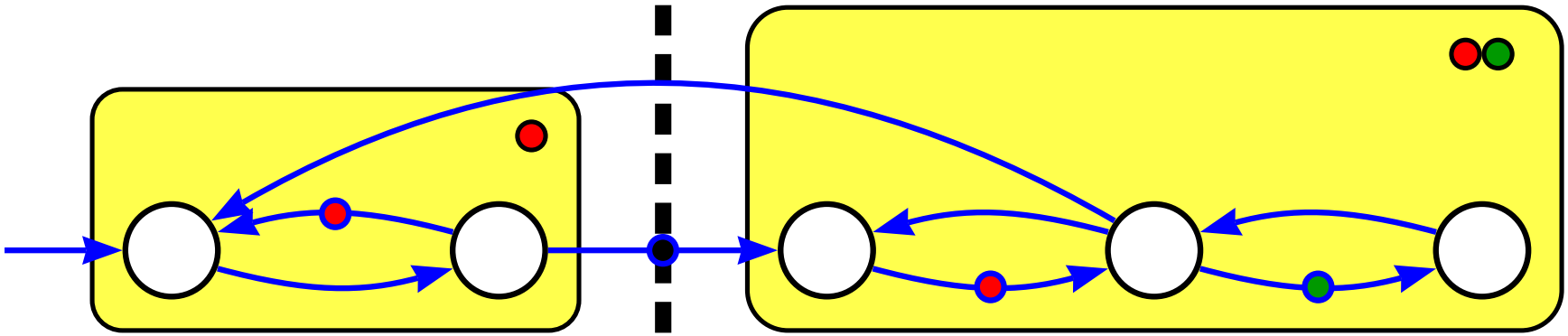
- Pour Büchi, on cherche des CFC dont les conditions d'acceptation vérifient $\bullet \wedge \circ \wedge \bullet \wedge \bullet$.
- Pour Streett, elle doivent vérifier $\bullet \Rightarrow \circ \wedge \bullet \Rightarrow \bullet$.



Comme il n'y a pas de $\circ$ dans cette CFC, on recommence, en considérant $\bullet$ comme une barrière à sens unique, interdisant les fusions de CFC.

Emptiness check pour automates de Streett

- Pour Büchi, on cherche des CFC dont les conditions d'acceptation vérifient $\bullet \wedge \circ \wedge \bullet \wedge \bullet$.
- Pour Streett, elle doivent vérifier $\bullet \Rightarrow \circ \wedge \bullet \Rightarrow \bullet$.



Comme il n'y a pas de $\circ$ dans cette CFC, on recommence, en considérant $\bullet$ comme une barrière à sens unique, interdisant les fusions de CFC.

Conclusion sur *emptiness check* pour Streett

Comparaison avec l'*emptiness check* pour GBA :

- Les CFC sont revisitées au pire m fois (si m paires de conditions d'acceptation).
- À comparer avec les $3^m + 1$ états de la traduction des formules LTL pour exprimer l'équité forte.

Ce nouvel algorithme permet de traiter l'équité forte avec un surcoût linéaire au lieu d'un surcoût exponentiel.

Conclusion générale : contributions

Intérêt des TGBA :

- La traduction gagne à étiqueter les transitions ;
- Les *emptiness checks* généralisés sont plus efficaces.
  (Si plus d'une condition d'acceptation ; p.ex. équité faible.)

Traduction LTL $\longrightarrow$ TGBA efficace :

- Améliorant et combinant des techniques existantes.

Comparatif et amélioration d'*emptiness checks* :

- Optimisations/heuristiques pour NDFS et CFC ;
- Algorithmes de calcul de contre-exemples ;

Nouveaux *emptiness checks* :

- Pour automates ensemblistes
- Pour automates de Streett (pour l'équité forte)

Spot : bibliothèque pour le *model checking*

`http://spot.lip6.fr`
C++, 70 000 lignes

- Fournit des briques pour réaliser des *model checkers*.

  Trois *model checkers* développés en interne :
  1. pour les réseaux de Petri ordinaires
  2. exploitant les symétries de réseaux de Petri colorés
  3. représentant les états de réseaux de Petri ordinaires par des BDD

- Bac à sable pour tester de nouveaux algorithmes (ou l'existant).

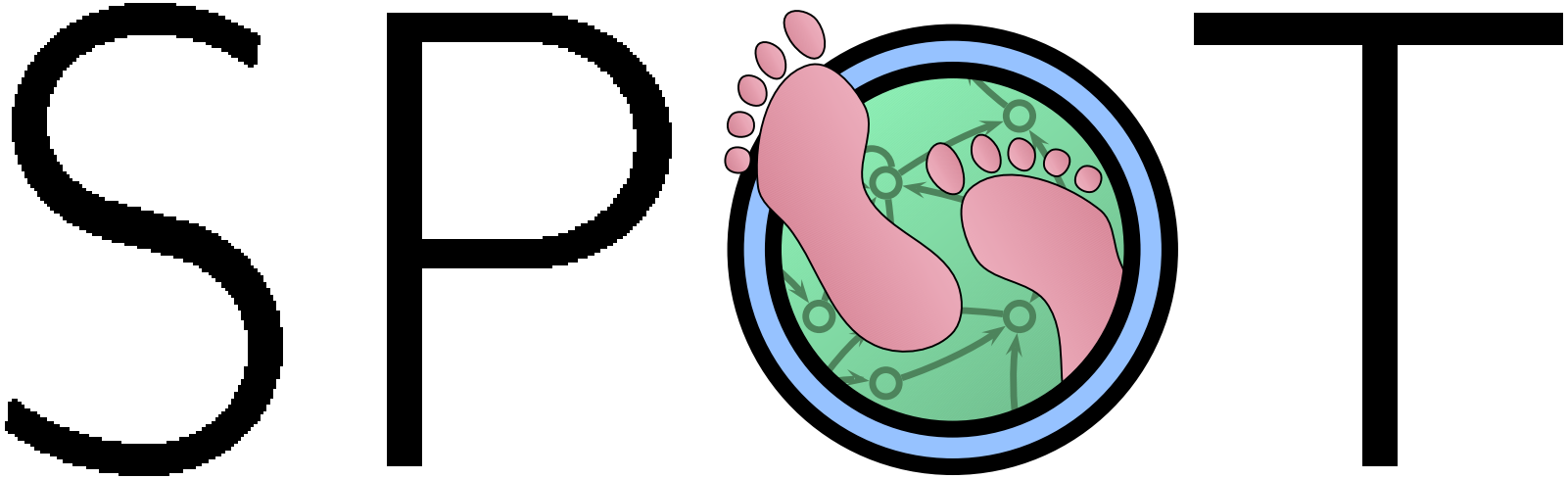
  Utilisé par
  - Tauriainen pour tester une heuristique ;
  - Li, Xie et Hao pour développer une variante de l'algorithme de Couvreur ;
  - Sebastiani, Tonetta et Vardi pour la traduction de formules LTL.

Perspectives

- Parallélisation de l'*emptiness check*
  - Pour Streett : réexploration d'une CFC dans un *thread* différent ;
  - En général : communication entre plusieurs processus qui cherchent à tester le même automate.
- Ensembles persistants équitables
  - Ensembles persistants pratiques à mettre en œuvre dans les *emptiness check* avec recherche de CFC ;
  - Traiter l'équité du modèle demande des définitions un peu plus fines.
- Existe-t-il une façon de traduire LTL en automate de Streett ?
  - On sait traduire

$$\varphi \wedge \left(\bigwedge_{i=1}^{n} \mathsf{G}\,\mathsf{F}\,a_i \rightarrow \mathsf{G}\,\mathsf{F}\,b_i \right)$$

Questions ?



« Spot Produces Our Traces »

http://spot.lip6.fr/

Bibliographie

Alexandre Duret-Lutz et Denis Poitrenaud.
Spot : an extensible model checking library using transition-based generalized Büchi automata.
In Proc. of MASCOTS'04, pages 76–83, Volendam, The Netherlands, October 2004. IEEE Computer Society Press.

Jean-Michel Couvreur, Alexandre Duret-Lutz et Denis Poitrenaud.
On-the-fly emptiness checks for generalized Büchi automata.
In Proc. of SPIN'05, volume 3639 of LNCS, pages 143–158. Springer-Verlag, August 2005.

Souheib Baarir et Alexandre Duret-Lutz.
Emptiness check of powerset Büchi automata.
In Proc. of ACSD'07, July 2007. To appear.