

Jeux sur des graphes infinis

Marne-la-Vallée,
Séminaire général,
22 Avril 2003

Thierry CACHAT
RWTH Aachen (Aix-la-Chapelle)

<http://www-i7.informatik.rwth-aachen.de/~cachat/>

Résumé

On définira les jeux à deux joueurs sur des graphes avec différentes conditions de gain (accessibilité, Büchi, parité) ainsi que les liens avec les questions de vérification (mu-calcul) et de synthèse de contrôleur pour les systèmes réactifs. Un algorithme "resout" un jeu s'il détermine quel joueur peut gagner à coup sûr et s'il calcule une stratégie gagnante. Sur les graphes finis, de tels algorithmes sont connus depuis plusieurs années. Le cas des graphes infinis (par exemple les graphes de transitions des automates à pile) est plus difficile. Plusieurs méthodes ont été développées récemment (Bouajjani et al, Vardi, Walukiewicz), dont nous exposerons les idées directrices.

Plan

- 1- jeux : définition
- 2- problèmes algorithmiques, vérification
- 3- graphes infinis : jeux à pile

Différentes méthodes de résolution :

- 4- méthodes symboliques,
- 5- jeu-simulation,
- 6- automate d'arbre

Références

E. Grädel, W. Thomas and T. Wilke eds.,
Automata, Logics, and Infinite Games,
A guide to current research, LNCS 2500, 2002.

projet "Games"

<http://www.games.rwth-aachen.de/index.html>

Jeu infini sur un graphe (fini)

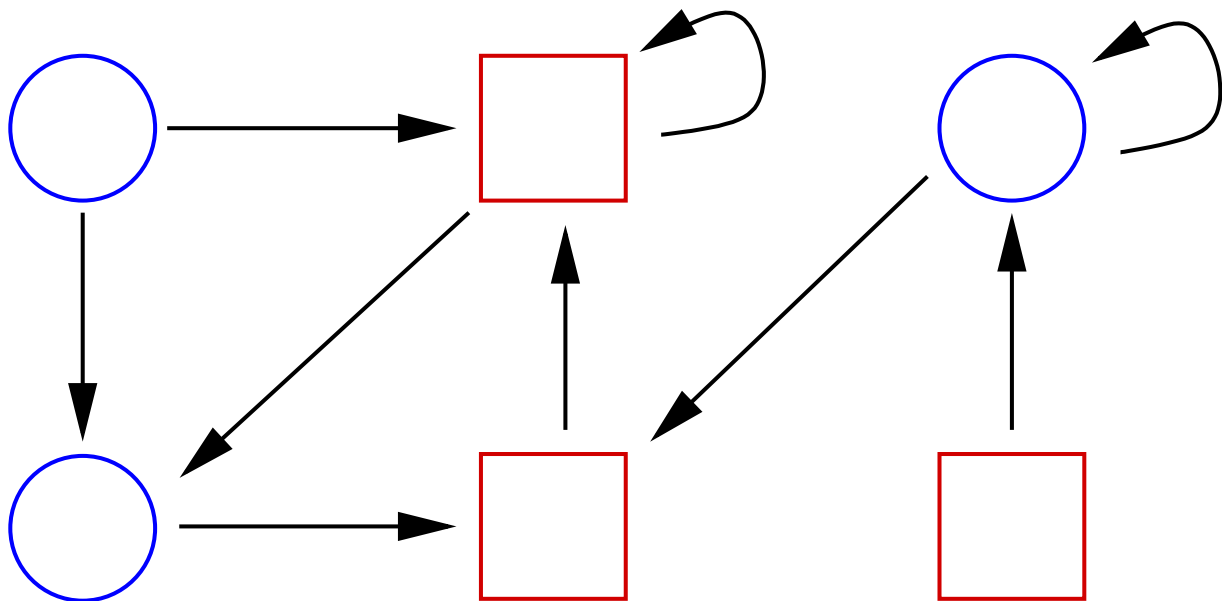
Arène : graphe orienté (V, E) , $E \subseteq V \times V$

$$V = V_0 \uplus V_1$$

Depuis un sommet v :

si $v \in V_0$, i.e. \bigcirc , **Joueur 0** choisit v' , vEv'

si $v \in V_1$, i.e. \square , **Joueur 1** choisit v' , vEv'



partie $\pi = \pi_0\pi_1 \cdots \in V^\omega$,

$$\forall i \pi_i E \pi_{i+1}$$

partie gagnée pour le Joueur 0 $\Leftrightarrow \pi \in L \subseteq V^\omega$

Différentes conditions de gain

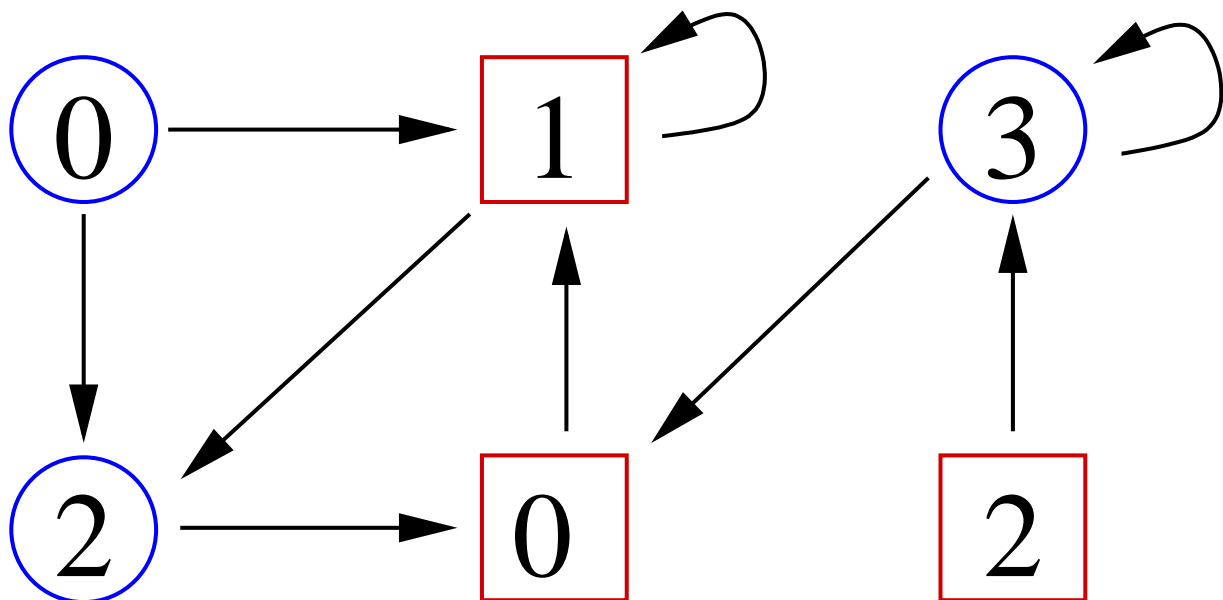
Accessibilité : étant donné $R \subseteq V$,

$$L = \{\pi \mid \exists i, \pi_i \in R\}$$

Récurrence (Büchi) : $L = \{\pi \mid \exists^\infty i, \pi_i \in R\}$

Parité : étant donné $c : V \rightarrow \{0, \dots, n\}$

$$L = \{\pi \mid \max(\text{Inf}(c(\pi))) \text{ est pair} \}$$



Problèmes algorithmiques

Depuis quelles positions le Joueur 0 peut-il gagner “à coup sûr” ?

Comment ?

Stratégie pour le Joueur 0

fonction (partielle) $f_0 : \begin{array}{l} V^*V_0 \longrightarrow V \\ \pi_0 \cdots \pi_n \longmapsto \pi_{n+1} \end{array}$

stratégie gagnante depuis une position : toutes les parties jouées conformément à la stratégie sont gagnantes pour le Joueur 0.

Cas particulier **stratégie de position**

$$f_0 : V_0 \longrightarrow V$$

Jeu déterminé.

Algorithmes connus pour les graphes finis.

Applications

Systèmes réactifs :

Contrôleur = Joueur 0

Environnement = Joueur 1

Vérification, synthèse de contrôleurs

Systeme fini S
formule ϕ du μ -calcul } \mapsto jeu (pol.)

$S \models \phi \iff \left\{ \begin{array}{l} \text{Joueur 0 a une} \\ \text{stratégie gagnante} \end{array} \right.$

[EJ 91], [EJS 93]

déterminer le gagnant *d'un jeu* de parité est $NP \cap co-NP$ [Jur 00].

Jeu à pile

Automate à pile (PDS) $\mathcal{P} = (P, \Gamma, \Delta)$

$P = P_0 \uplus P_1$ ensemble fini, “partagé”, d'états de contrôle,

Γ alphabet de pile, fini,

$\Delta \subseteq P \times \Gamma \times P \times \Gamma^*$ ensemble fini de règles de transition.

Graphe de transitions (V, \hookrightarrow)

$P\Gamma^* = V$ ensemble de sommets (configurations)

$\forall v \in \Gamma^*, p\gamma v \hookrightarrow qwv \Leftrightarrow (p, \gamma, q, w) \in \Delta$

Joueur 0 : $V_0 = P_0\Gamma^*$, Joueur 1 : $V_1 = P_1\Gamma^*$

condition de gain : accessibilité, Büchi, Parité, définies par l'état de contrôle

Jeu d'accessibilité

atteindre un ensemble R de “bonnes” configurations

Attracteur :

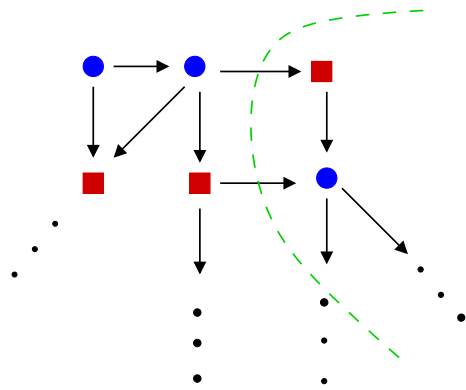
Région gagnante = $Attr(R) = \bigcup_{i \in \mathbb{N}} Attr^i$

$Attr^0 = R,$

$Attr^{i+1} = Attr^i$

$\cup \{v \in V_0 \mid \exists u, v \hookrightarrow u, u \in Attr^i\}$

$\cup \{v \in V_1 \mid \forall u, v \hookrightarrow u \Rightarrow u \in Attr^i\}.$



Approche symbolique : ensemble R rationnel reconnu par un automate fini \mathcal{A} . (alphabet $P \uplus \Gamma$)

Algorithme : ajouter successivement à \mathcal{A} des transitions pour construire \mathcal{A}_{Attr} , automate alternant qui reconnaît $Attr(R)$, **rationnel**.

[BH 1970] [BEM 97] [EHRS 00] [icalp 02]

ensemble rationnel de configurations

alphabet $P \uplus \Gamma$, automate fini...

en particulier : \mathcal{P} -automate

$$\mathcal{A} = (Q, \Gamma, \longrightarrow, P, F), \text{ tel que}$$

Q ensemble fini d'états,

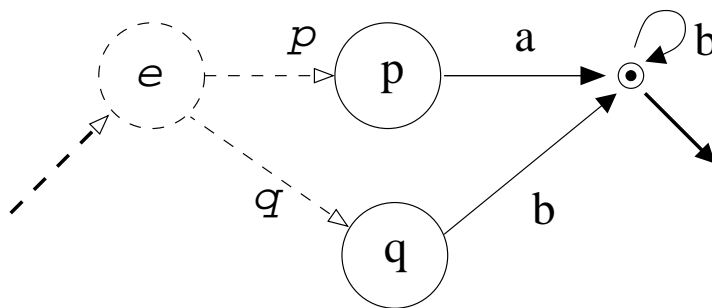
$\longrightarrow \subseteq Q \times \Gamma \times Q$ ensemble de transitions,

$P \subseteq Q$ ensemble des états initiaux : les états de contrôle de \mathcal{P} ,

$F \subseteq Q$ ensemble des états finaux.

Pas de transitions vers P .

$$R = \{pw \in P\Gamma^* \mid p \xrightarrow{w}^* f \in F\} \in \text{Rat}(P\Gamma^*)$$



$$pab^* \cup qb^+$$

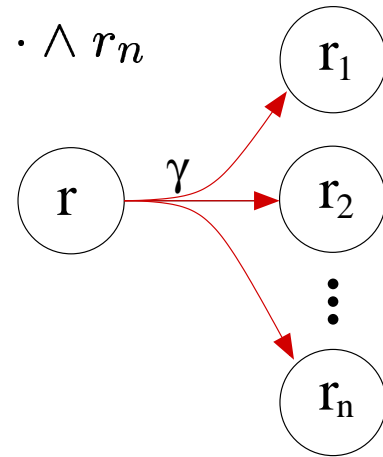
Automate alternant

\mathcal{A} est vu comme un automate **alternant** (EA)

transitions-ET :

$r \xrightarrow{\gamma} \{r_1, \dots, r_n\}$, avec $r, r_1, \dots, r_n \in Q$,

i.e. conjonction : $r \xrightarrow{\gamma} r_1 \wedge \dots \wedge r_n$



$$\longrightarrow^* \subseteq Q \times \Gamma^* \times 2^Q,$$

$$\forall r \in Q, r \xrightarrow{\epsilon}^* \{r\},$$

$$r \xrightarrow{\gamma} r' \Rightarrow r \xrightarrow{\gamma}^* \{r'\}$$

$$\left. \begin{array}{l} r \xrightarrow{\gamma} \{r_1, \dots, r_n\} \\ \forall i, r_i \xrightarrow{w}^* S_i \end{array} \right\} \Rightarrow r \xrightarrow{\gamma w}^* \bigcup_i S_i.$$

\mathcal{A} accepte le mot pw **ssi** il existe un chemin $p \xrightarrow{w}^* S$ avec $S \subseteq F$.

Ici $r \xrightarrow{\gamma} (r_1 \wedge r_2) \vee (r_3 \wedge r_4)$ représenté par **deux** transitions $r \xrightarrow{\gamma} \{r_1, r_2\}$ et $r \xrightarrow{\gamma} \{r_3, r_4\}$.

Algorithme

procédure de saturation

Répéter

- (Joueur 0) **si** $p \in P_0$, $p\gamma \hookrightarrow qw$ **et**
 $q \xrightarrow{w}^* S$ dans l'automate actuel,
alors ajouter la **nouvelle** transition $p \xrightarrow{\gamma} S$.

- (Joueur 1) **si** $p \in P_1$,

{	$p\gamma \hookrightarrow q_1 w_1$	sont toutes les transitions
	\vdots	
{	$p\gamma \hookrightarrow q_n w_n$	dans l'automate actuel,
	\vdots	

{	$q_1 \xrightarrow{w_1}^* S_1$	dans l'automate actuel,
	\vdots	
{	$q_n \xrightarrow{w_n}^* S_n$	dans l'automate actuel,
	\vdots	

alors ajouter la **nouvelle** transition $p \xrightarrow{\gamma} \bigcup_i S_i$.

jusqu'à aucune nouvelle transition ne peut être ajoutée.

$\rightsquigarrow \mathcal{A}_{Att}$, $L(\mathcal{A}_{Att}) = Attr(R)$

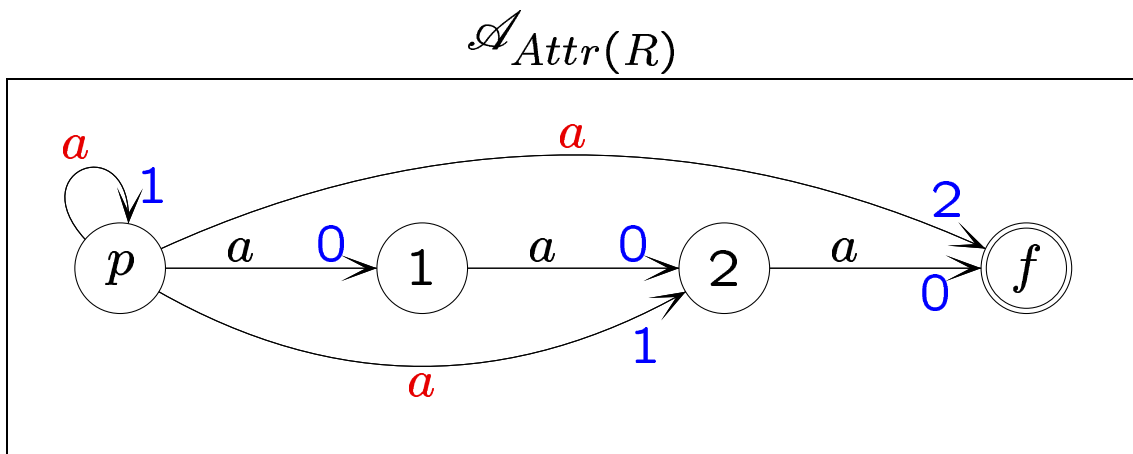
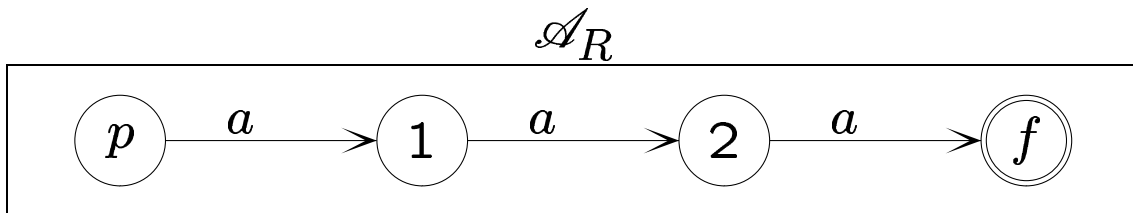
au plus $|\Gamma| \cdot |P| \cdot 2^{|Q|}$ transitions, temps $|\Delta| 2^{\mathcal{O}(|Q|^2)}$

[BH 1970] $\left(2^{2^{|Q|}}\right)^{2^{|Q|}}$.

Exemple de saturation

$$R = \{paaa\}, \quad P = P_0 = \{p\}, \quad P_1 = \emptyset, \quad \Gamma = \{a\}$$

$$\Delta = \{ (p, a, p, \varepsilon), (p, a, p, aa) \}$$



configuration paa acceptée par **deux** chemins **différents**, avec **les coûts 3 et 1** :

Coût minimal = coût de paa = 1

La stratégie choisit la “configuration suivante” qui a le plus petit coût.

Stratégie (pour le Joueur 0)

But : à une configuration $p\gamma u \in L(\mathcal{A}_{Attr})$ associer le coup suivant. Basée sur $\mathcal{A}_{Attr}(R)$.

idée : une “nouvelle” transition $p \xrightarrow{\gamma} S$ de $\mathcal{A}_{Attr}(R)$ associée à une unique règle $p\gamma \hookrightarrow qw$.
 \rightsquigarrow jouer vers $qwu \in Attr(R)$, MAIS, cela **ne** définit **pas** une stratégie gagnante : exemple

2 solutions

- définir des coûts (distance) \rightsquigarrow stratégie de position optimale,
- enregistrer les chemins \rightsquigarrow stratégie à pile.

Stratégie de position avec des coûts

But : depuis $pv \in Attr_0(R)$,
déterminer le $\min\{i \mid pv \in Attr_0^i\} = rang(pv)$.

coût d'une transition

Stratégie optimale.

Stratégie à pile

Automate à pile avec entrée/sortie, qui “lit” les coup du Joueur 1 et produit les coup du Joueur 0.

$\mathcal{S} = (P, A, \Pi)$, $A = \Gamma \times \Sigma$, Σ un ensemble fini

$$\Pi \subseteq (P_1 \times A \times \Delta_1) \times (P \times A^*) \cup (P_0 \times A) \times (P \times A^* \times \Delta_0)$$

configuration du jeu : $pw = p\gamma_0 \cdots \gamma_n$

et de la stratégie : $p(\gamma_0, \sigma_0) \cdots (\gamma_n, \sigma_n)$

exécution en temps constant

Jeux de Büchi

avec $Attr_0^+(X)$: atteindre X en au moins un coup.

$$\begin{aligned} Buechi_0^0(R) &= V = P\Gamma^*, \\ Buechi_0^{i+1}(R) &= Attr_0^+(Buechi_0^i(R) \cap R) \end{aligned}$$

$$Buechi_0(R) = \bigcap_{i \in \mathbb{N}} Buechi_0^i(R).$$

Première idée : Construire un automate \mathcal{B}^i qui reconnaît $Buechi_0^i(R)$

Mais, **pas de terminaison** : suite strictement décroissante de langages.

exm : $Buechi_0^i(R) = pa^i a^*$

Accélération

Idée de Simulation [Wal 96]

Étant donné un jeu de parité sur un graphe à pile (PDS), construire un jeu sur un graphe fini (FPG) “équivalent” :

Gagnant du PDS = Gagnant du FPG
Stratégie dans le PDS \longleftrightarrow Stratégie FPG
(partie dans le PDS ... \longleftrightarrow ... partie FPG)

Application :

Résoudre le FPG avec les algorithmes connus :
déterminer le gagnant, calculer une stratégie gagnante.

En déduire une stratégie gagnante pour le PDS.

[infinity 02]

Le jeu sur un graphe fini

(Version simplifié)

Une configuration $p\gamma v$ du PDS, $p \in P$, $\gamma \in \Gamma$, $v \in \Gamma^*$ est représentée dans le FPG par un sommet $Check(p, \gamma, B)$, où $B \subseteq P$ “résume” l’information contenue dans v :

$$B = \{q \in P \mid \text{Joueur 0 gagne depuis } qv\}.$$

opération “Pop” dans le PDS ($p\gamma v \hookrightarrow qv$)
 \Rightarrow gain ou perte immédiat dans le FPG, en fonction de B .

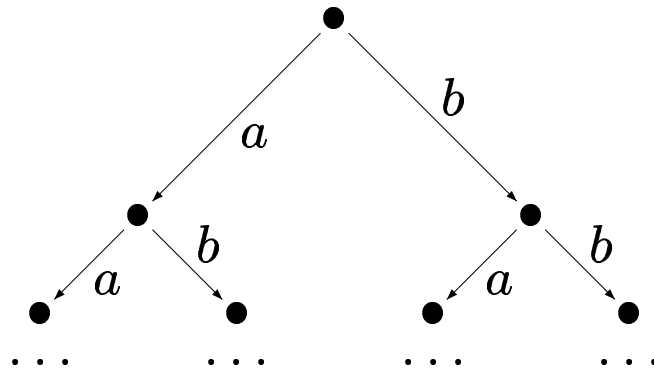
opération “Push” dans le PDS : le FPG passe à un sommet intermédiaire, où **Joueur 0** doit deviner ce qui arrivera plus tard :

Il choisit un $C \subseteq P$ tel que il prétend qu’après le prochain “pop”, le PDS sera dans un état de contrôle $q \in C$.

Joueur 1 a 2 possibilités : soit il relève le défi, soit il veut voir ce qui arrive après le “pop”.

Automate d'arbre [KV 00] [Var 98]

Arbre complet sur Γ : représente les contenus de pile.



Automate d'arbre bidirectionnel alternant simule le jeu (de parité).

Transformation en un automate unidirectionnel \rightsquigarrow problème du vide décidable, stratégie calculable.

[alig 02]

Logique monadique du second ordre

(MSO)

Exprime des propriétés sur les graphes :

“Il existe un sommet tel que ...”

jeu de parité \simeq μ -calcul $<$ MSO

\rightsquigarrow calcul de la région gagnante.

Pas la stratégie ?

[MS 85] : décidabilité de MSO sur les graphes des automates à pile.

Complexité.

Extensions

Autres conditions de gain : par exemple Σ_3
[csl 02]

Graphes préfixe-reconnaissables :

Alphabet Γ , langages rationnels U_i, V_i, W_i ,

sommets $V = \Gamma^*$, $E \subseteq \Gamma^* \times \Gamma^*$, $E =$

$$\{uw \leftrightarrow vw \mid u \in U_i, v \in V_i, w \in W_i, 1 \leq i \leq N\}$$

Jeux de parité [infinity 02]

Automates à pile de pile, jeux de parité : jeu-
réduction [icalp 03]

Autre classes de graphes... indécidabilité.

Références

- BEM 97** Ahmed Bouajjani, Javier Esparza, Oded Maler, *Reachability analysis of pushdown automata : Application to model-checking*, CONCUR '97, LNCS 1243, pp 135-150, 1997.
- BH 70** J. Richard Büchi and William H. Hosken, *Canonical systems which produce periodic sets* . Mathematical Systems Theory, 4-1, pp 81-90, 1970, or in *The collected works of J.Richard Büchi*, Saunders Mac Lane, Dirk Siefkes, 1990.
- icalp 02** Thierry Cachat, *Symbolic strategy synthesis for games on pushdown graphs*, ICALP'02, LNCS 2380, pp. 704-715, 2002.
- infinity 02** Thierry Cachat, *Uniform solution of parity games on prefix-recognizable graphs*, INFINITY 2002, ENTCS 68(6), 2002.
- alig 02** Thierry Cachat, *Two-way tree automata solving pushdown games*, in Automata, Logics, and Infinite Games, E. Grädel, W. Thomas and T. Wilke eds., ch. 17, pp. 303-317, LNCS 2500, 2002.
- csl 02** Thierry Cachat, Jacques Duparc and Wolfgang Thomas, *Solving Pushdown Games with a Σ_3 Winning Condition*, CSL'02, LNCS 2471, pp. 322-336, 2002.
- icalp 03** Thierry Cachat, *Higher Order Pushdown Automata, the Caucal Hierarchy of Graphs and Parity Games*, accepted at ICALP'03.

- EHRS 00** Javier Esparza, David Hansel, Peter Rossmanith and Stefan Schwoon, *Efficient algorithm for model checking pushdown systems*. Technische Universität München, 2000.
- EJ 91** E. Allen Emerson and Charanjit S. Jutla *Tree automata, mu-calculus and determinacy*, FoCS '91, IEEE Computer Society Press, pp. 368–377, 1991.
- EJS 93** E. Allen Emerson, Charanjit S. Jutla, and A. Prasad Sistla, *On model-checking for fragments of μ -calculus*, CAV '93, LNCS 697, pp. 385–396, 1993.
- Jur 00** Marcin Jurdziński, *Small progress measures for solving parity games*, STACS 2000, LNCS 1770, pp. 290–301, 2000.
- KV 00** Onar Kupferman und Moshe Y. Vardi, *An automata-theoretic approach to reasoning about infinite-state systems*, CAV'00, LNCS 1855, 2000.
- MS 85** David E. Muller and Paul E. Schupp, *The theory of ends, pushdown automata, and second-order logic*, TCS 37, 1985
- Var 98** Moshe Y. Vardi, *Reasoning about the past with two-way automata*, ICALP'98, vol. 1443 of LNCS, 1998
- Wal 96** Igor Walukiewicz, *Pushdown processes : games and model checking*, CAV'96, LNCS 1102, pp. 62–74, 1996. Full version in Information and Computation 164, pp. 234-263, 2001.