Références

[AN01] A. Arnold and D. Niwinski. *Rudiments of μ-calculus*, volume 146 of *Studies in Logic*. Elsevier, 2001.

[BVW94] O. Bernholtz, M. Vardi, and P. Wolper. Au automata-theoretic approach to branching-time model checking. In *Proc. 6th Int. Conf. on Computer Aided verification*, volume 818 of *Lecture Notes in Computer Science*. Springer Verlag, 1994.

[CES86] E. M. Clarke, E. A. Emerson, and A. P. Sistla. Automatic verification of finite-state concurrent systems using temporal logic specifications. *ACM Transactions on Programming Languages and Systems*, 8(2) :244–263, April 1986.

[Cle93] R. Cleaveland. A linear time model-checking algorithm for the alternation-free μ-calculus. *Formal methods in System Design*, 2, 1993.

[Dam94] Mads Dam. Temporal logic, automata and classical theories. Lecture notes for the sixth european summer school in logic, language and information, Copenhagen, 1994.

[DS02] Stéphane Demri and Philippe Schnoebelen. The complexity of linear temporal logic in simple cases. *Information and Computation*, 2002. To appear. Available at `http://www.lsv.ens-cachan.fr/Publis/publis-y3-2002.php`.

[EJ88] A. Emerson and C. Jutla. The complexity of tree automata and logics of programs. In *Proc. 29th IEEE Symposium on Foundations of Computer Science*, pages 368–377, 19988.

[Eme90] A. Emerson. *Handbook of Theoretical Computer Science, Vol. B*, chapter Temporal and Modal Logics. Elsevier, 1990.

[ES84] E.A. Emerson and A.P. Sistla. Deciding full branching time logic. *Information and Control*, 61 :175–201, 1984.

[EvW97] K. Etessami, M. vardi, and T. Wilke. First-order logic with two variables and unary temporal logic. In *Proc. IEEE Symposium on Logic in Computer Science*, Varsaw, 1997.

[GTW02] Erich Grädel, Wolfgang Thomas, and Thomas Wilke, editors. *Automata, Logics and Infinite Games*, volume 2500 of *Lecture Notes in Comp. Science*. Springer Verlag, 2002.

[Joh90] D.S. Johnson. *Handbook of Theoretical Computer Science*, volume A, chapter A Catalog of Complexity Classes, pages 67–162. North-Holland, 1990. J. van Leeuwen ed.

[Jur00] M. Jurdzinski. Small progress measures for solving parity games. In *Proc. Symp. Theoretical Aspects of Computer Science*, volume 1770 of *Lecture Notes in Comp. Science*. Springer Verlag, 2000.

[Koz83] D. Kozen. Results on the propositional μ-calculus. *Theoretical Comp. Science*, 27, 1983.

[KVW00] Orna Kupferman, Moshe Vardi, and Pierre Wolper. An automata-theoretic approach to branching time model-checking. *Journal of the ACM*, 47(2) :312–360, 2000.

[QS82] J. P. Queille and J. Sifakis. Specification and verification of concurrent systems in CESAR. In *Proc. 5th Int. Symp. on Programming, Turin, LNCS 137*, pages 337–351. Springer-Verlag, April 1982.

[SC85] A.P. Sistla and E.M. Clarke. The complexity of propositional linear time logic. *JACM*, 32, 1985.

[Sho77] J. R. Shoenfield. Axioms of set theory. In J. Barwise, editor, *Handbook of mathematical logic*, chapter B1, pages 321–344. North-Holland, 1977.

[Tho90] W. Thomas. Automata on infinite objects. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, pages 134–191. Elsevier, 1990.

[Tho97] Wolfgang Thomas. Languages, automata and logic. In G. Rozenberg and A. Salomaa, editors, *Handbook of Formal Languages*, volume 3, pages 389–456. Springer-Verlag, 1997.

[Tho99] Wolfgang Thomas. Complementation of Büchi automata revised. In Juhani Karhumäki, Hermann A. Maurer, Gheorghe Paun, and Grzegorz Rozenberg, editors, *Jewels are Forever, Contributions on Theoretical Computer Science in Honor of Arto Salomaa*, pages 109–120. Springer Verlag, 1999.

[Var96] M. Vardi. An automata-theoretic approach to linear time logic. In *Logic for concurrency : structure versus automata*, volume 1043 of *Lecture Notes in Comp. Science*. Springer Verlag, 1996.

[Var97] M. Vardi. Alternating automata :unifying truth and validity checking for temporal logics. In *Proc. 14th Conf. on Automated Deduction*, volume 1249 of *Lecture Notes in Computer Science*. Springer Verlag, 1997.

[Var98] M. Vardi. Reasoning about the past with two-way automata. In *Proc. Int. Conf. on Automata, Languages and Programming (ICALP)*, 1998.

[VS85] M. Vardi and L. Stockmeyer. Improved upper and lower bounds for modal logics of programs. In *Proc. 17th ACM Symposium on Theory of Computing*, pages 240–251, 1985.

[VW86a] M. Vardi and P. Wolper. An automata-theoretic approach to automatic program verification. In *Proc. 1st IEEE Symposium on Logic in Computer Science*, pages 222–231, 1986.

[VW86b] M. Vardi and P. Wolper. Automata theoretic techniques for modal logics of programs. *Journal of Computer and System Science*, 32, 1986.

[VW94] M. Vardi and P. Wolper. Reasoning about infinite computations. *Information and Computation*, 115(1), 1994.

[WD99] T. Wilke and M. Dickhöfer. The automata-theoretic method works for tctl model checking. In *Proc. ICALP 99, to appear*, 1999.

[Wil01] T. Wilke. Alternating tree automata, parity games and modal μ-calculus. *Bull. Belg. Math. Soc.*, 8 :357–388, 2001.

13 Solutions des exercices

Exercice 26 Supposons sans perdre de généralité que l'ensemble des rangs est $0..n$. Soit A_0^0 l'ensemble des états q de rang 0 tels que le joueur 1 peut forcer le retour sur q e n un nombre fini de coups. Cet ensemble est évidemment un ensemble de positions gagnantes pour le joueur 1. A_0^1 est l'ensemble des états desquels le joueur 1 peut forcer à aller dans un état de A_0^0 en un nombre fini de coups. Plus généralement, soit A_{2k}^0 l'ensemble des états q tels que le joueur 1 peut forcer à revenir sur q sans passer par un état de rang inférieur à $2k$ et A_1^1 l'ensemble des états desquels le joueur 1 peut forcer à atteindre un état de $\bigcup_k A_{2k}^0$ en un nombre fini de coups. A_2^1 est l'ensemble des états desquels le joueur 1 peut forcer en un nombre fini de coups à atteindre l'un des états de A_1^1 (NB : A_2^1 peut contenir strictement A_1^1 dans le cas par exemple où le joueur 2 n'a le choix qu'entre aller dans un état de A_2^0 ou dans un état de A_0^0). Plus généralement, soit A_n^1 l'ensemble des états desquels le joueur 1 peut forcer à atteindre A_{n-1}^1 en un nombre fini de coups. Soit A_0^2 l'union des A_n^1.

Si l'état initial est dans A_0^2, le joueur 1 a une stratégie gagnante sans mémoire. Sinon, le joueur 2 a une stratégie permettant d'éviter tous les états de A_{2k}^0, pour tout k. Or, puisqu'il n'y a qu'un nombre fini d'états, dans une partie gagnante pour le joueur 1, il existe un état de rang pair par lequel on passe infiniment souvent sans passer par un état de rang inférieur. Autrement dit, si le joueur 1 a une stratégie gagnante, dans toute partie, on atteint un jour un état de A_1^1. Ainsi, si l'état initial n'est pas dans A_0^2, il existe une stratégie gagnante pour le joueur 2.

Exercice 25 Soit l'automate (le rang est l'indice des états) : $\delta(q_2,0) = q_2, \delta(q_2,1) = q_1, \delta(q_1,0) = q_2, \delta(q_1,1) = q_1$. L'automate accepte les mots qui ne contiennent qu'un nombre fini de 1.

L'automate obtenu par application du lemme 30 a pour transitions et états accessibles :

δ'	0	1
$\{q_2\},\emptyset$	$\emptyset,\{q_2\}$	$\{q_1\},\emptyset$
$\emptyset,\{q_2\}$	$\{q_2\},\emptyset$	$\{q_1\},\emptyset$
$\{q_1\},\emptyset$	$\{q_2\},\emptyset$	$\{q_1\},\emptyset$

Il accepte $(0010)^\omega$ qui contient une infinité de 1.