Model Checking Branching Time Logics

Ph. Schnoebelen
LSV, ENS Cachan & CNRS
61 av Pdt. Wilson, F-94230 Cachan, France
phs@lsv.ens-cachan.fr

Branching-time logics are temporal logics that allow quantification over possible futures [3, 4]. Such logics have been considered very early by the automated verification community because efficient model-checking algorithms for logics like CTL could be easily implemented and were used successfully [1].

When L is a (pure-future) linear-time logic, one obtains a branching-time logic $B(L)$ by extending L with quantification over branches. Thus CTL^* is $B(LTL)$ and CTL is $B(LTL_1)$, where we let LTL_1 denote the subset of LTL formulae having at most one occurrence of a temporal modality. Using a model-checking algorithm for L as an oracle, one directly obtains a model-checking algorithm for $B(L)$. Hence, if the model-checking problem for L has complexity $\mathcal{C}$, the model-checking problem for $B(L)$ is in $\mathrm{P}^{\mathcal{C}}$ [5, 9].

These observations are particularly illuminating for branching-time logics $B(L)$ where L has a NP-complete model-checking problem [2]. This is the case for some well-known logics, like CTL^+ or $FCTL$ [6], but also for some more expressive logics like BTL_2 [8] or $B^*(\mathsf{X})$ [10]. In such cases, the model-checking problem for $B(L)$ is in P^{NP}, i.e., the level Δ_2^p of the polynomial-time hierarchy, and it is of course both NP-hard and coNP-hard.

For these logics, the problem of closing the gap between these upper and lower bounds was left open until recent years and required the development of new techniques. In one direction, lower bounds tighter than NP- and coNP-hardness have been proved with new reductions from nested satisfiability problems [6, 10, 7]. In the other direction, new algorithms have been developed to solve model-checking problems with a limited number of invocations of the NP oracle [10].

References

[1] E. M. Clarke, E. A. Emerson, and A. P. Sistla. Automatic verification of finite-state concurrent systems using temporal logic specifications. *ACM Trans. Prog. Lang. Syst.*, 8(2):244–263, 1986.

[2] Ph. Demri, S. Schnoebelen. The complexity of propositional linear temporal logics in simple cases. *Information and Computation*, 174(1):84–103, 2002.

[3] E. A. Emerson and J. Y. Halpern. Decision procedures and expressiveness in the temporal logic of branching time. *J. Comput. Syst. Sci.*, 30(1):1–24, 1985.

[4] E. A. Emerson and J. Y. Halpern. "Sometimes" and "Not Never" revisited: On branching versus linear time temporal logic. *J. ACM*, 33(1):151–178, 1986.

[5] E. A. Emerson and C.-L. Lei. Modalities for model checking: Branching time logic strikes back. *Science of Computer Programming*, 8(3):275–306, 1987.

[6] F. Laroussinie, N. Markey, and Ph. Schnoebelen. Model checking CTL^+ and $FCTL$ is hard. In *Proc. 4th Int. Conf. Foundations of Software Science and Computation Structures (FOSSACS 2001), Genova, Italy, Apr. 2001*, volume 2030 of *Lecture Notes in Computer Science*, pages 318–331. Springer, 2001.

[7] F. Laroussinie, N. Markey, and Ph. Schnoebelen. Efficient timed model checking for discrete-time systems. *Theoretical Computer Science*, 353(1–3):249–271, 2006.

[8] A. Rabinovich and Ph. Schnoebelen. BTL_2 and the expressive power of $ECTL^+$. *Information and Computation*, 204(7):1023–1044, 2006.

[9] Ph. Schnoebelen. The complexity of temporal logic model checking. In *Advances in Modal Logic, vol. 4, selected papers from 4th Conf. Advances in Modal Logic (AiML 2002), Sep.-Oct. 2002, Toulouse, France*, pages 437–459. King's College Publication, 2003.

[10] Ph. Schnoebelen. Oracle circuits for branching-time model checking. In *Proc. 30th Int. Coll. Automata, Languages, and Programming (ICALP 2003), Eindhoven, NL, July 2003*, volume 2719 of *Lecture Notes in Computer Science*, pages 790–801. Springer, 2003.