Computational Complexity of Propositional Dynamic Logics

Summary of Dissertation

Stefan Göller

Fakultät für Mathematik und Informatik

Universität Leipzig

goeller@informatik.uni-leipzig.de

In 1979, Fischer and Ladner introduced Propositional Dynamic Logic (PDL) as a logical formalism for reasoning about programs [9]. Since then, PDL has become a classic of logic in computer science [11], and many extensions and variations have been proposed. Several of these extensions are inspired by the original application of reasoning about programs, while others aim at the numerous novel applications that PDL has found since its invention. Notable examples of such applications include agent-based systems [14], regular path constraints for querying semi-structured data [2], and XML-querying [1, 17, 16]. In artifical intelligence, PDL received attention due to its close relationship to description logics [10] and epistemic logic [18, 19].

The models of PDL formulas and programs are transition systems (also called *Kripke structures*) whose transitions are labeled with atomic programs and whose states are labeled with atomic propositions. *Formulas* are defined from atomic propositions, are closed by the boolean operations, and finally for each program π and each formula φ, $\langle\pi\rangle\varphi$ is a again a formula – with the semantics that there is some state both satisfying φ and reachable by executing π. Hence, formulas define subsets of the state set of Kripke structures. *Programs* are built up from atomic ones, are closed under the operations union, composition, and Kleene star, and finally whenever φ is a formula, then $\varphi?$ is a program defining loops in those states that satisfy φ. Hence programs define binary relations on the state set of Kripke structures.

The *satisfiability problems* asks, given a formula φ, whether there exists a Kripke structure K and a state of K that satisfies φ.
The *model checking problem* asks, given a Kripke structure K, a state x of K, and a formula φ, whether x satisfies φ.

In this thesis we investigate the computational complexity of the satisfiability problem and of the model checking problem over infinite state systems of fragments and of extensions of PDL. Our first results are concerned with an extension of PDL that allows intersection and converse on programs and also the usage of fixed points and nominals. Nominals are atomic propositions that hold in precisely one state of Kripke structures. We call the resulting logic μ-ICPDL$_{\texttt{Nom}}$. We define an operator # on pointed Kripke structures – these are Kripke structures K together with a state of K – and show that applying # to any pointed Kripke structure K yields a pointed Kripke structure $K^{\#}$ of bounded tree width (a widespread notion from graph theory) satisfying the same μ-ICPDL$_{\texttt{Nom}}$ formulas.

Theorem 1. *Let M be a finite set of nominals. Then for every pointed Kripke structure K over M we have that $K^{\#}$ has tree width at most $|M|+2$ and moreover K and $K^{\#}$ satisfy the same μ-ICPDL$_{\texttt{Nom}}$ formulas containing nominals from M.*

Next, we introduce a special reasoning problem for μ-ICPDL$_{\texttt{Nom}}$ that we call ω-regular tree satisfiability. Formally, *ω-regular tree satisfiability* is the question, given a two-way alternating parity tree automaton over infinite trees $\mathcal{T}$ (see also [21]) and a μ-ICPDL$_{\texttt{Nom}}$ formula φ that is interpreted over the same trees as $\mathcal{T}$, to decide whether there exists some tree T accepted by $\mathcal{T}$ such that φ holds in the root of T. We prove that ω-regular tree satisfiability in μ-ICPDL$_{\texttt{Nom}}$ is complete for 2EXP. As a first application of ω-regular tree satisfiability we reduce general satisfiability in μ-ICPDL$_{\texttt{Nom}}$ to it. This is done by an appropriate encoding of all countable Kripke structures of bounded tree width as an ω-regular tree language. We introduce an appropriate notion of nestings of the intersection operator in formulas and programs which the complexity of (ω-regular tree) satisfiability is doubly exponential in. We call the latter measure *intersection width*.

Theorem 2. *Satisfiability in μ-ICPDL$_{\texttt{Nom}}$ is* 2EXP*-complete. For every constant $c \geq 1$ satisfiability of μ-ICPDL$_{\texttt{Nom}}$ formulas of intersection width at most c is complete for* EXP*.*

For μ-ICPDL$_{\texttt{Nom}}$'s fragment ICPDL, i.e. PDL extended with intersection and converse, the previously best known upper bound for satisfiability was non-elementary [12].

Next, we investigate infinite state model checking of (fragments of) μ-ICPDL$_{\texttt{Nom}}$. We consider basic process algebras (BPA), pushdown systems (PDS) and prefix-recognizable systems (PRS). Aside from the natural class of prefix-recognizable systems (introduced by Caucal [5]), we follow Mayr's classification of infinite state systems. The reason for blinding out systems of Mayr's process rewrite systems [13] that involve parallel composition is that already model checking test-free PDL over basic parallel processes (also known as communication-free nets) is undecidable. We show the latter via a reduction from the undecidable model checking problem of CTL's fragment EF over Petri nets [7, 8]. Following Vardi [20], we distinguish three ways of measuring the complexity of the model checking problem: Data complexity (the formula is fixed, the Kripke structure is the input), expression complexity (the Kripke structure is fixed, the formula is the input) and combined complexity (both the formula and the Kripke structure are part of the input).

Again, via a simple reduction to ω-regular tree satisfiability we can prove an upper bound for infinite state model checking.

Theorem 3. *Model checking μ-ICPDL$_{\texttt{Nom}}$ over prefix-recognizable systems is in* 2EXP*. Moreover, for every constant $c \geq 1$ model checking μ-ICPDL$_{\texttt{Nom}}$ formulas of intersection width at most c over prefix-recognizable systems is* EXP*-complete.*

We can also show that the previous 2EXP upper bound is tight.

Theorem 4. *There exists already a fixed basic process algebra $\mathcal{X}$ such that model checking test-free IPDL over $\mathcal{X}$ is* 2EXP*-hard.*

The results for infinite state model checking of (test-free) IPDL/μ-ICDPL over BPA, PDS, and PRS are summarized in Table 1.

Next, we study an extension of PDL that allows to reason about dynamically changing Kripke structures. We investigate the complexity of satisfiability in $\mathrm{DLP}^+_{\mathrm{dyn}}$, introduced in [6]. Via a natural satisfiability preserving translation to PDL, Demri proved that satisfiability in $\mathrm{DLP}^+_{\mathrm{dyn}}$ is in 2EXP [6]. For $\mathrm{DLP}^+_{\mathrm{dyn}}$'s fragment $\mathrm{DLP}_{\mathrm{dyn}}$ of Pucella and Weissman, the best known upper bound was NEXP [15]. We close these complexity gaps by showing that satisfiability in $\mathrm{DLP}^+_{\mathrm{dyn}}$ is complete for EXP. The starting point for proving this EXP upper bound is a proposal by Demri from [6] that suggests to translate $\mathrm{DLP}^+_{\mathrm{dyn}}$ formulas to formulas of an extension of PDL, called PDL$\oplus$, that allows a certain operator $\oplus$ on programs. Hence, for

		BPA/PDS/PRS
IPDL, μ-ICPDL$_{\texttt{Nom}}$ constant intersection width	**data**	EXP-complete
	expression	
	combined	
μ-ICPDL$_{\texttt{Nom}}$	**data**	
	expression	2EXP-complete
	combined	

Table 1: Complexity of infinite state model checking of μ-ICPDL$_{\texttt{Nom}}$

proving an EXP upper bound for $\mathrm{DLP}^{+}_{\mathrm{dyn}}$, the hope would be to prove an EXP upper bound for PDL$\oplus$. We show that this approach may not be successful by proving that satisfiability in PDL$\oplus$ is complete for 2EXP. However, we syntactically restrict the $\oplus$ operator in a subtle way – this leads to the logic that we call PDL$\oplus[\mathbb{A}]$. It is worth mentioning that Demri's translation from $\mathrm{DLP}^{+}_{\mathrm{dyn}}$ to PDL$\oplus$ does not yield formulas of PDL$\oplus[\mathbb{A}]$. Summarizing, we prove EXP membership of satisfiability in $\mathrm{DLP}^{+}_{\mathrm{dyn}}$ in two steps. In a first step, we give a satisfiability preserving translation from $\mathrm{DLP}^{+}_{\mathrm{dyn}}$ to PDL$\oplus[\mathbb{A}]$. In a second step, we prove that satisfiability in PDL$\oplus[\mathbb{A}]$ is in EXP.

Theorem 5. *Satisfiability in DLP^{+}_{dyn} is* EXP*-complete.*

Finally, for the sake of completeness, we investigate the complexity of infinite state model checking of test-free and full PDL via reductions from known results of infinite state model checking of EF, CTL and the modal μ-calculus, respectively. The results are summarized in Table 2. We emphasize the following quite surprising lower bound result.

Theorem 6. *Reachability on prefix-recognizable systems is* EXP*-hard.*

This EXP lower bound matches the complexity of model checking the μ-calculus over prefix-recognizable systems [4]. Too, the EXP lower bound is somewhat surprising, since the complexity of *alternating* reachability over prefix-recognizable systems is EXP-complete as well. Note that for most graph classes there is a complexity jump from the reachability problem to the alternating reachability problem. Furthermore, EXP-completeness of reachability on prefix-recognizable systems stands in contrast to the rather low complexity (P-completeness) of reachability on pushdown systems [3].

References

[1] L. Afanasiev, P. Blackburn, I. Dimitriou, B. Gaiffe, E. Goris, M. Marx, and M. de Rijke. PDL for ordered trees. *Journal of Applied Non-Classical Logics*, 15(2):115–135, 2005.

[2] N. Alechina, S. Demri, and M. de Rijke. A modal perspective on path constraints. *Journal of Logic and Computation*, 13(6):939–956, 2003.

[3] A. Bouajjani, J. Esparza, and O. Maler. Reachability analysis of pushdown automata: Application to model-checking. In *Proceedings of the 8th International Conference on Concurreny*, volume 1243 of *Lecture Notes in Computer Science*, pages 135–150. Springer, 1997.

[4] T. Cachat. Uniform solution of parity games on prefix-recognizable graphs. *Electronic Notes Theoretical Computer Science*, 68(6), 2002.

		BPA	**PDS**	**PRS**
PDL\?	**data**	P-complete		EXP -complete
	expression		PSPACE-complete	
	combined			EXP-complete
PDL	**data**	P-complete		
	expression		EXP-complete	
	combined			

Table 2: Complexity of infinite state model checking (test-free) PDL.

[5] D. Caucal. On infinite transition graphs having a decidable monadic theory. *Theoretical Computer Science*, 290(1):79–115, 2002.

[6] S. Demri. A reduction from DLP to PDL. *Journal of Logic and Computation*, 15(5):767–785, 2005.

[7] J. Esparza. On the Decidabilty of Model Checking for Several mu-calculi and Petri Nets. In *Proceedings of the 19th International Colloquium on Trees in Algebra and Programming*, volume 787 of *Lecture Notes in Computer Science*, pages 115–129. Springer, 1994.

[8] J. Esparza. Decidability of model checking for infinite-state concurrent systems. *Acta Informatica*, 34(2):85–107, 1997.

[9] M. J. Fischer and R. E. Ladner. Propositional dynamic logic of regular programs. *Journal of Computer and System Sciences*, 18(2):194–211, 1979.

[10] G. De Giacomo and M. Lenzerini. Boosting the Correspondence between Description Logics and Propositional Dynamic Logics. In *Proceedings of the 12th National Conference on Artifical Intelligence*, pages 205–212, 1994.

[11] D. Harel, D. Kozen, and J. Tiuryn. *Dynamic Logic*. Foundations of computing. The MIT Press, 2000.

[12] C. Lutz. PDL with Intersection and Converse is Decidable. In *Proceedings of the Annual Conference for Computer Science Logic*, volume 3624 of *Lecture Notes in Computer Science*. Springer Verlag, 2005.

[13] R. Mayr. Decidability of model checking with the temporal logic EF. *Theoretical Computer Science*, 256(1-2):31–62, 2001.

[14] J. Meyer. Dynamic Logic for Reasoning about Actions and Agents. In Jack Minker, editor, *Logic-Based Artificial Intelligence*, pages 281–311. Kluwer Academic Publishers, 2000.

[15] R. Pucella and V. Weissman. Reasoning about Dynamic Policies. In *Proceedings of the 7th International Conference on Foundations of Software Science and Computation Structures*, volume 2987 of *Lecture Notes in Computer Science*, pages 453–467. Springer, 2004.

[16] B. ten Cate. The expressivity of XPath with transitive closure. In *Proceedings of the 25th ACM Symposium on Principles of Database Systems*, pages 328–337. ACM Press, 2006.

[17] B. ten Cate and C. Lutz. The Complexity of Query Containment in Expressive Fragments of XPath 2.0. In *Proceedings of the 26th ACM Symposium on Principles of Database Systems*, pages 73–82. ACM Press, 2007.

[18] W. van der Hoek and J. Meyer. A complete epistemic logic for multiple agents - combining distributed and common knowledge. In *Epistemic Logic and the Theory of Games and Decisions*, pages 35–68. Kluwer, 1997.

[19] H. P. van Ditmarsch, W. van der Hoek, and B. P. Kooi. Concurrent dynamic epistemic logic for MAS. In *Proceedings of the 2nd International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 201–208. ACM Press, 2003.

[20] M. Y. Vardi. The Complexity of Relational Query Languages. In *Proceedings of the 14th Annual ACM Symposium on Theory of Computing*, pages 137–146. ACM Press, 1982.

[21] M. Y. Vardi. Reasoning about The Past with Two-Way Automata. In *Proceedings of the 25th International Colloquium on Automata, Languages and Programming*, volume 1443, pages 628–641, London, UK, 1998. Springer.