Towards a Description Logic for Program Analysis: Extending $\mathcal{ALCQIO}$ with Reachability*

Tomer Kotek, Mantas Šimkus, Helmut Veith, and Florian Zuleger

Vienna University of Technology
{kotek,veith,zuleger}@forsyte.at, simkus@dbai.tuwien.ac.at

Shape analysis attempts to analyze and verify correctness of programs with dynamic data structures. This is a notoriously difficult task, because it necessitates efficient decision procedures for expressive logics on graphs and graph-like structures. In the last decade, model-theoretic approaches have been less prominent, and the leading approach is proof-theoretic [15]. Recent advances in finite model theory have created an opportunity for development of practical model-theoretic approaches in shape analysis.

Description Logics (*DLs*) are a well established family of logics for Knowledge Representation and Reasoning [2]. They model the domain of interest in terms of *concepts* (classes of objects) and *roles* (binary relations between objects). These features make DLs very useful to formally describe and reason about graph-structured information. The usefulness of DLs is witnessed e.g. by the W3C choosing DLs to provide the logical foundations to the standard Web Ontology Language (OWL) [14]. Another application of DLs is formalization and static analysis of UML class diagrams and ER diagrams, which are basic modeling artifacts in object-oriented software development and database design, respectively [4,1]. In these settings, standard reasoning services provided by DLs can be used to verify e.g. the consistency of a diagram.

To describe the memory of programs with dynamic data structures using a DL, a rather powerful DL must be chosen. The DL in question needs to allow a computationally problematic combination of constructors: (i) nominals are required to represent the program's variables; (ii) number restrictions are required so that the program's pointers (represented as roles) are interpreted as functions; (iii) inverses are needed for defining data structures such as trees, where elements in the tree must have at most one parent, and for encoding program computation; and (iv) reachability is required since data structures should contain only elements which are reachable from program variables via program pointers.

Our contribution: We introduce and develop decision procedures for the logic $\mathcal{ALCQIO}_{b,Re}$, which extends the closure of $\mathcal{ALCQIO}$ under Boolean operations ($\mathcal{ALCQIO}_b$) with reachability assertions over finite structures. The main

* The first, third and fourth author were supported by the Austrian National Research Network S11403-N23 (RiSE) of the Austrian Science Fund (FWF) and by the Vienna Science and Technology Fund (WWTF) through grants PROSEED and ICT12-059. The second author was supported by the FWF project P25518 and the WWTF project ICT12-15.

results of this paper are algorithms which decide the finite satisfiability and finite implication problems of $\mathcal{ALCQIO}_{b,Re}$. The algorithms are reductions to finite satisfiability in $\mathcal{ALCQIO}$, which suggests relatively simple implementation using existing $\mathcal{ALCQIO}$ reasoners. Currently no $\mathcal{ALCQIO}$ reasoners over finite structures exist to our knowledge. Still, reasoners over arbitrary structures could be used as part of a program verification procedure which is sound but not complete. The algorithms run in NEXPTIME, which is optimal since $\mathcal{ALCQIO}$ is already NEXPTIME-hard. We only consider finite structures since only they and not infinite structures can represent the memory of programs.

The reachability assertions guarantee that elements of the universe of a model are reachable in the graph-theoretic sense from initial sets of elements using prescribed sets of binary relation symbols. Alternatively, we can think of $\mathcal{ALCQIO}_{b,Re}$ as $\mathcal{ALCQIO}_b$ interpreted over structures containing an unbounded number of trees of bounded degree d.

$\mathcal{ALCQIO}_{b,Re}$ is obtained from $\mathcal{ALCQIO}$ by (1) allowing the Boolean connectives $\vee, \wedge, \neg$ and (2) adding two new types of assertions:

Reachability Assertion $B \overrightarrow{\subseteq}^S A$ where $A, B \in \mathsf{N_C}$ and $S \subseteq \mathsf{N_F}$. Intuitively, it says that B is contained in A and that A is a set of elements reachable from B, without leaving A, through the roles of S.

Disjointness Assertion $Disj(A_1, A_2) = (A_1 \sqcap A_2 \equiv \bot)$ for $A_1, A_2 \in \mathsf{N_C}$.

Let RE and DI be sets of reachability respectively disjointness assertions.

Compatibility RE and DI *compatible* if for every $B_1 \overrightarrow{\subseteq}^{S_1} A_1$ and $B_2 \overrightarrow{\subseteq}^{S_2} A_2$ in RE such that $S_1 \cap S_2 \neq \emptyset$, $Disj(A_1, A_2)$ is in DI.

If we think of the A_i as the sets of elements in different data structures in the memory, then compatibility is the natural statement that that data structures can only share the same domain if they use different pointers.

Let $*$ denote the reflexive-transitive closure, $\circ$ denote role composition, $R^{\mathcal{M}} = (\bigcup_{s \in S} s^{\mathcal{M}} \cap A^{\mathcal{M}} \times A^{\mathcal{M}})^*$ and $R^{\mathcal{M}}(B^{\mathcal{M}}) = \{v \mid \exists (u, v) \in R^{\mathcal{M}}. u \in B^{\mathcal{M}}\}$. We have $\mathcal{M} \models B \overrightarrow{\subseteq}^S A$ iff $\mathcal{M} \models B \sqsubseteq A$ and $R^{\mathcal{M}}(B^{\mathcal{M}}) = A^M$.

Theorem 1. *Let $\Phi_i \in \mathcal{ALCQIO}_{b,Re}$ for $i = 1, 2$. There are polynomial-time computable $\mathcal{ALCQIO}$ formulas η and ρ over an extended vocabulary such that*

(1) Φ_1 is satisfiable iff η is satisfiable.
(2) Φ_1 implies Φ_2 iff ρ is not satisfiable.
(3) Satisfiability and implication in $\mathcal{ALCQIO}_{b,Re}$ is NEXPTIME-complete.

Satisfiability and implication here are over finite structures.

The proof of Theorem 1 is by reduction to the satisfiability problem of $\mathcal{ALCQIO}$. The models of $\Phi = \Phi_1$ resp. $\Phi = \Phi_1 \rightarrow \Phi_2$ can be partitioned into *standard* and *non-standard models*, depending on whether they satisfy the reachability assertions. Since $\mathcal{ALCQIO}$ is contained in first order logic, $\mathcal{ALCQIO}$ cannot express the reachability assertions. However, we can augment Φ so that it is guaranteed that whenever a non-standard model exists, so does a standard model, and

the standard model can be obtained from the non-standard model by means of so-called *tree surgery*. The construction relies on a locality property of $\mathcal{ALCQIO}$.

The logic $\mathcal{ALCQIO}_{b,Re}$ is especially suited to shape analysis, since it contains nominals, number restrictions, inverses and reachability. $\mathcal{ALCQIO}_{b,Re}$ is strong enough to describe e.g. lists, trees and lists of lists. $\mathcal{ALCQIO}_{b,Re}$ supports programs whose data structures have complex sharing patterns, and memory cells (which in model-theoretic terms are elements of the universe of the model) may participate in multiple data structures. The closure of the underlying logic $\mathcal{ALCQIO}_b$ under Boolean operations allows to describe conditional statements in programs. The decision procedure for implication for $\mathcal{ALCQIO}_{b,Re}$ is essential for verification applications, since it allows to show that specifications relating pre- and post-conditions are correct.

Since $\mathcal{ALCQIO}_{b,Re}$ is a DL, using $\mathcal{ALCQIO}_{b,Re}$ for shape analysis brings an additional advantage. The verification community has focused mostly on a bottom-up approach to the analysis of programs with dynamic data structures, which examines pointers and the shapes induced by them. However, many real world programs manipulate complex data whose structure and content is most naturally described by formalisms from object oriented programming and databases such as UML and ER diagrams which are generalized by the framework of description logic. In another extended abstract in this volume (see also [8]) we discusse how to use a DL to reason and verify correctness of entity-relations-type content of data structures on top of an existing shape analysis.

Related work

We list some DLs with some form of reachability from the literature. The important work of Schild [17] exposed a correspondence between variants of propositional dynamic logic (PDL), a logic for reasoning about program behavior, and variants of DLs extended with further role constructors, e.g. the transitive closure of a role. Close correspondences between DLs extended with fixpoints and variants of the μ-calculus have also been identified [18,12,5,13,16,6]. Extensions of DLs with regular expressions over roles have been proposed [7]. $\mathcal{ALCQIO}$ is the extension of $\mathcal{ALC}$ with nominals, number restrictions and inverses, see e.g. [3]. No extensions of $\mathcal{ALCQIO}$ with reachability or transitive closure were known to be decidable on finite structures. [10] extended $\mathcal{SHOIQ}$ with transitive closure of roles and proved decidable in non-deterministic triple exponential time on arbitrary structures. Our result has a similarity in proof strategy with a recent decidability result for an extension of the two-variable fragment of first order logic with trees and counting [9]. Our results are incomparable with [9].

The use of DLs in shape analysis has been previously suggested in [11], where a framework for verification is given based mainly on the description logics $\mu\mathcal{ALCQIO}$, which extends $\mathcal{ALCQIO}$ with fixed points, and on $\mu\mathcal{ALCQO}$. However, unlike $\mathcal{ALCQIO}_{b,Re}$, from the methods of [6] it follows that $\mu\mathcal{ALCQIO}$ is undecidable over finite structures, and $\mu\mathcal{ALCQO}$ is unknown to be decidable on finite structures.

References

1. A. Artale, D. Calvanese, R. Kontchakov, V. Ryzhikov, and M. Zakharyaschev. Reasoning over extended ER models. In *Conceptual Modeling-ER 2007*, volume 4801 of *LNCS*, pages 277–292. Springer, 2007.
2. F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, and P. F. Patel-Schneider, editors. *The Description Logic handbook: theory, implementation, and applications*. Cambridge University Press, 2003.
3. F. Baader, C. Lutz, M. Milicic, U. Sattler, and F. Wolter. Integrating description logics and action formalisms: First results. In *AAAI*, pages 572–577, 2005.
4. D. Berardi, D. Calvanese, and G. De Giacomo. Reasoning on UML class diagrams. *Artificial Intelligence*, 168(1–2):70–118, 2005.
5. Piero A. Bonatti, Carsten Lutz, Aniello Murano, and Moshe Y. Vardi. The complexity of enriched mu-calculi. *Logical Methods in Computer Science*, 4(3), 2008.
6. Piero A. Bonatti and Adriano Peron. On the undecidability of logics with converse, nominals, recursion and counting. *Artif. Intell.*, 158(1):75–96, 2004.
7. D. Calvanese, T. Eiter, and M. Ortiz. Regular path queries in expressive description logics with nominals. In *IJCAI-09*, pages 714–720, 2009.
8. D. Calvanese, T. Kotek, M. Simkus, H. Veith, and F. Zuleger. Shape and content: Incorporating domain knowledge into shape analysis. *CoRR*, abs/1312.6624, 2013.
9. W. Charatonik and P. Witkowski. Two-variable logic with counting and trees. In *LICS*, pages 73–82, 2013.
10. Chan Le Duc, Myriam Lamolle, and Olivier Curé. A decision procedure for shoiq with transitive closure of roles. In *International Semantic Web Conference (1)*, pages 264–279, 2013.
11. L. Georgieva and P. Maier. Description logics for shape analysis. In *SEFM*, pages 321–330. IEEE, 2005.
12. G. De Giacomo and M. Lenzerini. Concept language with number restrictions and fixpoints, and its relationship with mu-calculus. In *ECAI*, pages 411–415, 1994.
13. Orna Kupferman, Ulrike Sattler, and Moshe Y. Vardi. The complexity of the graded μ-calculus. In *CADE*, pages 423–437, 2002.
14. W3C OWL Working Group. *OWL 2 Web Ontology Language: Document Overview*. W3C Recommendation, 27 October 2009. Available at `http://www.w3.org/TR/owl2-overview/`.
15. J. C. Reynolds. Separation Logic: A logic for shared mutable data structures. In *LICS*, pages 55–74, Washington, DC, USA, 2002. IEEE Computer Society.
16. Ulrike Sattler and Moshe Y. Vardi. The hybrid μ-calculus. In *IJCAR*, pages 76–91, 2001.
17. K. Schild. A correspondence theory for terminological logics: Preliminary report. In *IJCAI*, pages 466–471, 1991.
18. K. Schild. Terminological cycles and the propositional μ-calculus. In *KR*, pages 509–520. Morgan Kaufmann, 1994.