Shape and Content

Incorporating Domain Knowledge into Shape Analysis*

D. Calvanese[1], T. Kotek[2], M. Šimkus[2], H. Veith[2], and F. Zuleger[2]

[1] Free University of Bozen-Bolzano
[2] Vienna University of Technology

Abstract. The verification community has studied dynamic data structures primarily in a bottom-up way by analyzing pointers and the shapes induced by them. Recent work in fields such as separation logic has made significant progress in extracting shapes from program source code. Many real world programs however manipulate complex data whose structure and content is most naturally described by formalisms from object oriented programming and databases. In this paper, we attempt to bridge the conceptual gap between these two communities. Our approach is based on Description Logics (DLs), a widely used knowledge representation paradigm which gives a logical underpinning for diverse modeling frameworks such as UML and ER. We show how DLs can be used on top of an existing shape analysis to add content descriptions to the shapes. Technically, we assume that we have separation logic shape invariants obtained from a shape analysis tool, and requirements on the program data in terms of description logic. We show that the two-variable fragment of first order logic with counting and trees (whose decidability was presented at LICS 2013) can be used as a joint framework to embed suitable DLs and separation logic.

1 Introduction

The manipulation and storage of complex information in imperative programming languages is often achieved by dynamic data structures. The verification of programs with dynamic structures however is notoriously difficult, and is a highly active area of current research. This paper aims to put a new perspective on this problem. We discuss how the analysis of the shape can be complemented by an analysis of the content to be stored.

Shape analysis is concerned with the analysis of pointers and of the structures induced by them. Recent years have seen considerable progress in automatic methods for inferring basic shape properties such as lists and trees and variations thereof (cyclic lists, doubly-linked lists, etc.). This success has been enabled by succinct formalisms for representing heap structures, most notably separation

* Kotek, Veith and Zuleger 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. Šimkus was supported by the FWF grants P25518 and the WWTF grants ICT12-15.

logic [12,7]. With a few exceptions e.g. [9,10] the majority of papers on shape analysis has focused on the (graph-theoretic) shape of the data structures rather than their information content. For instance, classical shape analysis does not capture simple concepts such as "a list of students where each student has a list of teachers" but only the more combinatorial concept "list of lists".

Content representation has been studied by several disciplines including databases, modeling and knowledge representation. These research communities typically model reality by classes and binary relationships between these classes. For example, the database community uses entity-relationship (ER) diagrams, and UML diagrams have been studied in requirements engineering. UML and ER diagrams can often be expressed in *Description Logics (DLs)*, which have been studied extensively by the knowledge representation community, and is used for many modeling tasks, see [2].

Content representation in the form of UML and ER has become a central pillar of industrial software engineering. In complex software projects, the source code is usually accompanied by design documents which provide extensive documentation and models of data structure content. This documentation is both an opportunity and a challenge for program verification. Recent hardware verification papers have demonstrated how design diagrams can be integrated into an industrial verification workflow [8].

Problem Statement and Main Goal. Shape analysis and content representation have been very successful paradigms for their respective scope and purpose. They show their limitations, however, when we consider programs which lie in their intersection. This large class of programs is using the heap extensively to store complex information: essentially, the heap is an *in-memory database*. In line with the discussion above, this database is often an informal instantiation of an abstract UML or ER scheme.

The goals of verification are often related to *both* the content and the data structure representation, e.g. the verification of complex heap invariants in a file system, the adherence to security policies, etc. Classical shape analysis, however, is not equipped to represent relationships between complex data on the appropriate abstraction level; mutatis mutandis, suitable content representation formalisms such as DLs are oblivious of the data structures and operations to manage the data on the heap.

We conclude that shape analysis and content representation represent two valid views of the heap, (i) as a collection of data structures satisfying structural and graph-theoretic properties in the former case, and (ii) as a collection of containers of entities and their relationships in the latter case. In order to adequately analyze programs with dynamically allocated data structures, it is necessary to *combine* the two approaches. In this paper, we choose a DL and a separation logic fragment as concrete logical frameworks to study this question.

The methodological goal of this paper is a proof of concept to establish description logic as an assertion and proof formalism for the verification of high-level properties of programs with dynamic data structures. The technical challenge is to find a verification methodology along with a suitable formalism that

bridges the gap between shape analysis and content representation, i.e., between description logic and separation logic.

Logics. DLs are mature and well understood logics, they have good algorithmic properties and enjoy efficient reasoners. DLs vary in expressivity and complexity, and are usually selected according to the expressivity needed to formalize the given target domain. A precise framework for reasoning over UML class diagrams and ER diagrams can be found in [3,1]. Moreover, DLs are the logical backbone of the Web Ontology Language (OWL) for the Semantic Web [11]. In this paper we employ a very expressive description logic (henceforth called $\mathcal{L}$), based on the so called $\mathcal{ALCHOIF}$, which we specifically tailor to better support reasoning about complex pointer structures.

Separation logic is a powerful proof-theoretic framework which is used to reason about correctness of programs with dynamically allocated memory. Separation logic is an extension of Hoare logic. Due to its support for local reasoning, separation logic is the most prominent logic for reasoning about the heap. While early papers on separation logic [12] dealt with highly expressive but undecidable logics, we use a fragment of separation logic from [4] which was successfully used in program analysis tools and whose reasoning complexity is polynomial.

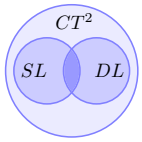

Fig. 1.

In order to study the combination of description logic and separation logic, we identify a powerful decidable logic which incorporates both. In Figure 1, SL and DL denote the separation logic fragment and the description logic that we use, respectively. The logic CT^2 is an extension of first order logic with counting and trees. Motivated by applications in shape analysis, a recent deep result in [6] shows that finite satisfiability of CT^2-formulae is in NEXPTIME. Thus, [6] connects a major line of research in finite model theory to shape and content analysis.

Our contributions:

- The semantics of SL is given in terms of heap functions. We define *memory structures* based on DL semantics for representing the heap and transform the semantics of SL to a semantics based on memory structures.
- We study the description logic $\mathcal{L}$ as a formalism for expressing *content properties* of memory structures using concrete examples.
- We give an embedding of a fragment of the separation logic from [4] into CT^2. $\mathcal{L}$ has a fairly standard reduction (see e.g. [5]) to CT^2. Moreover, we give a complexity-preserving reduction of satisfiability of CT^2 over memory structures to finite satisfiability of CT^2.
- We describe a program model for sequential imperative heap-manipulating programs without procedures. Our first main contribution is a Hoare-style proof system for verifying content properties on top of (already verified) shape properties stated in separation logic.
- Our second main contribution is precise backward-translation of content properties along loop-less code. This backward-translation allows us to reduce the inductiveness of the Hoare-annotations to satisfiability in CT^2. We prove the soundness and completeness of this reduction.

References

1. A. Artale, D. Calvanese, R. Kontchakov, V. Ryzhikov, and M. Zakharyaschev. Reasoning over extended ER models. In *Proc. of ER*, 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. D. Berardi, D. Calvanese, and G. De Giacomo. Reasoning on UML class diagrams. *Artificial Intelligence*, 168(12):70 – 118, 2005.
4. J. Berdine, C. Calcagno, and P.W. O'Hearn. Symbolic execution with Separation Logic. In *APLAS*, volume 3780, pages 52–68. Springer-Verlag, 2005.
5. A. Borgida. On the relative expressiveness of description logics and predicate logics. *Artif. Intell.*, 82(1-2):353–367, 1996.
6. W. Charatonik and P. Witkowski. Two-variable logic with counting and trees. In *LICS*, pages 73–82, 2013.
7. S. S. Ishtiaq and P. W. O'Hearn. Bi as an assertion language for mutable data structures. POPL, pages 14–26. ACM, 2001.
8. D. James, T. Leonard, J. O'Leary, M. Talupur, and M. R. Tuttle. Extracting models from design documents with mapster. PODC, 2008.
9. S. Magill, J. Berdine, E. Clarke, and B. Cook. Arithmetic strengthening for shape analysis. In *Static Analysis*, volume 4634 of *LNCS*, pages 419–436. Springer, 2007.
10. S. Magill, M.-H. Tsai, P. Lee, and Y.-K. Tsay. Automatic numeric abstractions for heap-manipulating programs. POPL, 2010.
11. W3C OWL Working Group. *OWL 2 Web Ontology Language: Document Overview*. W3C Recommendation, 27 October 2009.
12. J. C. Reynolds. Separation Logic: A logic for shared mutable data structures. In *In Proc. of LICS*, pages 55–74, Washington, DC, USA, 2002. IEEE Computer Society.