Shape and Content*
A Database-Theoretic Perspective
on the Analysis of Data Structures

Diego Calvanese\textsuperscript{1}, Tomer Kotek\textsuperscript{2}, Mantas Šimkus\textsuperscript{2}, Helmut Veith\textsuperscript{2}, and Florian Zuleger\textsuperscript{2}

\textsuperscript{1} Free University of Bozen-Bolzano
\textsuperscript{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 look at the verification of programs with dynamic data structures from the perspective of content representation. Our approach is based on description logic, a widely used knowledge representation paradigm which gives a logical underpinning for diverse modeling frameworks such as UML and ER. 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 can be used as a joint framework to embed suitable fragments of description logic 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 data structures, however, is notoriously difficult, and is a highly active area of current research. While much progress has been made recently in analyzing and verifying the shape of dynamic data structures, most notably by separation logic (SL) \cite{24,17}, the content of dynamic data structures has not received the same attention.

In contrast, disciplines as databases, modeling and knowledge representation have developed highly-successful theories for content representation and verification. These research communities typically model reality by classes and binary

\footnote{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. Simkus was supported by the FWF grant P25518 and the WWTF grant ICT12-15.}
relationships between these classes. For example, the database community uses entity-relationship (ER) diagrams, and UML diagrams have been studied in requirements engineering. 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 [18].

In this paper, we propose the use of Description Logics (DLs) for the formulation of content specifications. DLs are a well established and highly popular family of logics for representing knowledge in artificial intelligence [3]. In particular, DLs allow to precisely model and reason about UML and ER diagrams [6,2]. DLs are mature and well understood, they have good algorithmic properties and have efficient reasoners. DLs are very readable and form a natural base for developing specification languages. For example, they are the logical backbone of the Web Ontology Language (OWL) for the Semantic Web [22]. DLs vary in expressivity and complexity, and are usually selected according to the expressivity needed to formalize the given target domain.

Unfortunately, the existing content representation technology cannot be applied directly for the verification of content specifications of pointer-manipulating programs. This is due to the strict separation between high-level content descriptions such as UML/ER and the way data is actually stored. For example, query languages such as SQL and Datalog provide a convenient abstraction layer for formulating data queries while ignoring how the database is stored on the disk. In contrast, programs with dynamic data structures manipulate their data structures directly. Moreover, database schemes are usually static while a program may change the content of its data structures over time.

The main goal of this paper is to develop a verification methodology that allows to employ DLs for formulating and verifying content specifications of pointer-manipulating programs. We propose a two-step Hoare-style verification methodology: First, existing shape-analysis techniques are used to derive shape invariants. Second, the user strengthens the derived shape invariants with content annotations; the resulting verification conditions are then checked automatically. Technically, we employ a very expressive DL (henceforth called $\mathcal{L}$), based on the so called $\text{ALCHOIF}$, which we specifically tailor to better support reasoning about complex pointer structures. For shape analysis we rely on the SL fragment from [7]. In order to reason automatically about the verification conditions involving DL as well as SL formulae, we identify a powerful decidable logic $\text{CT}^2$ which incorporates both logics [10]. We believe that our main contribution is conceptual, integrating these different formalisms for the first time. While the current approach is semi-manual, our long term goal is to increase the automatization of the method.