Automated Higher-order Reasoning in Quantales

Han-Hing Dang

Institut für Informatik, Universität Augsburg, D-86135 Augsburg, Germany
`h.dang@informatik.uni-augsburg.de`

Abstract. We present an approach to bring reasoning in quantales into the realm of automated theorem proving. We use the TPTP Problem Library for this purpose which by a recent approach now integrates fully automated higher-order theorem provers. In particular, we give an encoding of quantales in the new typed higher-order form and show how to prove theorems about quantales fully automatically.

1 Introduction

Nowadays automated theorem provers have found wide spread applications in the area of formal verification due to an immense increase of computing capabilities over the last decades. Especially algebraic structures, such as Kleene algebras and relation algebras, have proven to be very effective for lots of verification tasks of different areas. Furthermore also quantales are of special interest and can be easily extended to Kleene Algebras. For example, quantales were used to describe the algebraic semantics of the temporal logics CTL*, CTL and LTL [10]. Up to now only Kleene algebras and relation algebras have the advantage that they come with a simple first-order equational calculus which directly allows fully automated first-order theorem provers to be applied [9]. Contrarily, axiomatisations of quantales mostly require higher-order encodings. For quite some time it was assumed that theorem provers for higher-order logic need a lot of user interaction and therefore are highly non-automatic. Hence this approach was not considered for automated reasoning. Due to a recent approach by G. Sutcliffe and others for such systems, well known higher-order theorem provers such as *Isabelle* [11], *LEO II* [2] and *TPS* [1] were now integrated into the *TPTP Problem Library* [13] with already hundreds of higher-order problems [3]. These systems are running fully automated using various search strategies or by interaction with first-order theorem provers for first-order fragments.

In this paper we show how quantales can be encoded into the typed higher-order form of the TPTP library and list some theorems which we have proved fully automatically using that approach.

2 Quantales

In this section we give definitions for quantales. Their axiomatisations are of main interest in this paper.

A *quantale* [12] is a structure $(S, \leq, 0, \cdot, 1)$ where $(S, \leq)$ is a complete lattice and $\cdot$ is a completely disjunctive inner operation on S, i.e., $\cdot$ distributes over arbitrary suprema. Moreover 0 is the least element of the lattice w.r.t. $\leq$ and 1 is the identity of multiplication. The infima of arbitrary sets $X \subseteq S$ are denoted by $\bigsqcap X$ while the suprema are denoted by $\sum X$. The binary variants for two elements $a, b \in S$ are written as $a \sqcap b$ and $a + b$, resp. Furthermore $\top =_{df} \sum S$ is used to denote the greatest element of S.

In particular, the notion of a quantale is equivalent to Conway's notion of a *standard Kleene algebra* [7] and a special idempotent semiring. We will have a closer look on the axiomatisation of standard Kleene algebras in the following. According to Conway, a standard Kleene algebra is axiomatically defined as a set S with three operations $\sum$, $\cdot$, * and elements 0, 1. We consider for quantales only the operations $\sum$ and $\cdot$ that satisfy

$$\begin{array}{cc}
\sum \emptyset = 0\,, & \sum\{x\} = x\,, \\
\sum \bigcup_{i \in I}\{\sum X_i\} = \sum \bigcup_{i \in I} X_i\,, & x \cdot 1 = 1 \cdot x = x\,, \\
(x \cdot y) \cdot z = x \cdot (y \cdot z)\,, & \sum X_1 \cdot \sum X_2 = \sum\{x_1 \cdot x_2 : x_1 \in X_1, x_2 \in X_2\},
\end{array}$$

assuming X_1, $X_2 \subseteq S$, $X_i \subseteq S$ for all $i \in I \subseteq \mathbb{N}$ and x, y, $z \in S$. We will use these definitions of $\sum$ and $\cdot$ in the following to encode quantales in the typed higher-order TPTP library.

3 THF0 and Church's Simple Type Theory

The recent higher-order approach in the TPTP problem library implements higher-order logic in the so-called THF0 core by Church's simple theory of types [6] since this theory is already used as a common basis for a lot of higher-order ATP systems[1] [4]. Church's simple type theory is based on the simply typed λ-calculus in which functional types are formed from basic types. These consists of individuals `$i`, Boolean values `$o` and further types using the function type constructor `>`. All symbols are written in THF0 syntax. Special types can be defined in THF0. The encoded annotated formulas we use in this paper are all of the form

```
thf( <formula_name>, <role>, ( <formula> )).
```

Detailed examples will be shown in the next section while `<formula_name>` and `<role>` should be clear. The occurring symbols `!`, `?` and `^` stand for $\forall$, $\exists$ and λ, resp. The binary operator `@` denotes function application.

4 Encoding in Higher-order TPTP-Syntax

As it can be seen from Section 2 in Conway's axiomatisation of quantales, a suitable encoding of set-theory in higher-order logic is required. We have chosen

[1] ATP is used here and in the remainder for automated theorem proving

an encoding that was also proposed by Benzmüller et al. [4]. This encoding has already been successfully implemented and used for proving lots of theorems. The basic idea is to represent sets by their characteristic functions. In particular, we use for an element x and a set X the following equivalence

$$x \in X \Leftrightarrow X(x).$$

By this, set operations such as intersection or union can be expressed very naturally and easily using the typed λ-calculus notation. Binary union e.g. can be defined by $\lambda X, Y, x. ((X\,x) \vee (Y\,x))$ assuming that X, Y have the type $\alpha \to \{true, false\}$ and x has the type α.

In the following we only give an extract of the complete input file to demonstrate the encoding. A full version can be found online [8]. We only consider the supremum definition here since it forms the most interesting part of the encoding.

```
thf(sup,type,(
    sup: ( ( $i > $o ) > $i ) )).
```

This formula defines the type of the supremum operation. It takes a characteristic function of an arbitrary set as an argument, denoted by `( $i > $o )` and returns its supremum of type `$i`. With this type definition we can give the first two axioms:

```
thf(sup_es,axiom,( (sup @ emptyset) = zero )).

thf(sup_singleset,axiom,(
    ! [X: $i] : ( ( sup @ ( singleton @ X ) ) = X ) )).
```

Clearly the function `emptyset` maps every element of type `$i` into false. Furthermore in the second formula `! [X: $i]` denotes a $\forall$-quantification over all elements `X` and `( singleton @ X )` a set containing only a single element `X`.

For the axiom $\sum \bigcup_{i \in I} \{\sum X_i\} = \sum \bigcup_{i \in I} X_i$ we have to model functions that represent the sets $\bigcup_{i \in I} \{\sum X_i\}$ and $\bigcup_{i \in I} X_i$. This is done by defining functions that take a set of sets as an argument. The type definitions should be clear but are given in the following for completeness.

```
thf(supset,type,(
    supset: ( ( ( $i > $o ) > $o ) > $i > $o ) )).

thf(unionset,type,(
    unionset: ( ( ( $i > $o ) > $o ) > $i > $o ) )).
```

The function `supset` represents the set $\bigcup_{i \in I} \{\sum X_i\}$ and `unionset` represents the set $\bigcup_{i \in I} X_i$. They are defined as follows:

```
thf(supset,definition,
    ( supset
    = ( ^ [F: ( $i > $o ) > $o, X: $i ] :
      ? [Y: $i > $o] : ( ( F @ Y ) & ( ( sup @ Y ) = X ) ) ) )).

thf(unionset,definition,
    ( unionset
    = ( ^ [F: ( $i > $o ) > $o, X: $i ] :
      ( ? [Y: $i > $o] : ( ( F @ Y ) & ( Y @ X ) ) ) ) )).

thf(sup_set,axiom,(
    ! [X: ( $i > $o ) > $o] : ( ( sup @ ( supset @ X ) ) =
                                ( sup @ ( unionset @ X ) ) ) )).
```

Arbitrary index sets I can now be handled by quantification over sets of sets. This is a big advantage of higher-order encodings. The `sup_set` axiom above would be very complex and very difficult to read if expressed in first-order logic. Especially set theory is encoded much more naturally in higher-order logic. Moreover Benzmüller et al. stated that theorems are solved more efficiently than using first-order encodings [5]. Even more, we have the benefit that, due to its uniformity, every higher-order theorem prover can now be used that supports the THF0 syntax.

5 Results

In this section we present the results from our encoding of Section 4. We proved a lot of theorems in quantales, in particular theorems using infinitary sums. Hence automated theorem proving can also be applied to quantales.

The encoding of the $\sum$ operation given in Section 4 might give the impression that the introduction of the existentially quantified set variables `Y` will lead to bad results due to a large increase of the state space. However, from this encoding it was possible to prove a lot of interesting theorems automatically. The first experience we got was that the automatic version of Isabelle performed best for these exercises. We list some of the proved theorems in the following:

$$\begin{array}{cc}
\sum\emptyset = \sum\{0\} & \sum\{x,0\} = x \\
\sum\{\sum\{x,y\},z\} = \sum\{x,\sum\{y,z\}\} & \sum(X\cup\{0\}) = \sum X \\
x \in X \;\Rightarrow\; x \le \sum X & Y \subseteq X \;\Rightarrow\; \sum Y \le \sum X \\
x\cdot 0 = 0\cdot x = 0 & \sum X\cdot 0 = 0\cdot \sum X = 0 \\
x\cdot(y+z) = x\cdot y + x\cdot z & (y+z)\cdot x = y\cdot x + z\cdot x
\end{array}$$

assuming S is a quantale whereas x, y, $z \in S$ and X, $Y \subseteq S$. The order $x \le y$ is defined by $x + y = y$ and $x + y =_{df} \sum\{x, y\}$ for arbitrary x, y.

6 Conclusion and Outlook

We presented an approach to bring the algebraic structure of quantales into the realm of automated reasoning. This was done by using an higher-order approach for ATP systems. In particular we presented an encoding from which it was possible to prove several interesting theorems about quantales. For example all remaining axioms of semirings or isotony w.r.t. infinitary sums can be proved fully automatically.

In this paper we used Conway's axiomatisation of quantales. It would be interesting to investigate more suitable axiomatisations and and more efficient encodings for the THF0 core since difficult theorems still need extra lemmas for full automation as first-order ATP systems. Therefore also higher-order problems as presented here will be integrated into the TPTP library to improve higher-order ATP systems for reasoning in algebraic structures as quantales.

Acknowledgements: I thank Bernhard Möller, Peter Höfner, Geoff Sutcliffe and Chad Brown for fruitful discussions and remarks.

References

1. P. B. Andrews and C. E. Brown. TPS: A Hybrid Automatic-interactive System for Developing Proofs. *Journal of Applied Logic*, 4(4):367–395, 2006.
2. C. Benzmüller, L. Paulson, F. Theiss, and A. Fietzke. The LEO-II project. In *Proceedings of the Fourteenth Workshop on Automated Reasoning, Bridging the Gap between Theory and Practice*. Imperial College, London, England, 2007.
3. C. Benzmüller, F. Rabe, A. v. Gelder, G. Sutcliffe, and C. Brown. Typed Higher-order Form. `http://www.cs.miami.edu/~tptp/TPTP/Proposals/THF.html`, 2009.
4. C. Benzmüller, F. Rabe, and G. Sutcliffe. THF0 — The Core of the TPTP Language for Higher-order Logic. In *Ijcar '08: Proceedings of the 4th international Joint Conference on Automated Reasoning*, pages 491–506. Springer, 2008.
5. C. Benzmüller, V. Sorge, M. Jamnik, and M. Kerber. Combined reasoning by automated cooperation. *Journal of Applied Logic*, 6(3):318–342, 2008.
6. A. Church. A formulation of the simple theory of types. *Journal of Symbolic Logic*, 5:56–68, 1940.
7. J. H. Conway. *Regular Algebra and Finite Machines*. Chapman & Hall, 1971.
8. P. Höfner. Database for automated proofs of Kleene algebra. `http://www.kleenealgebra.de`.
9. P. Höfner and G. Struth. Automated reasoning in Kleene algebra. In F. Pfenning, editor, *Automated Deduction — CADE-21*, volume 4603 of *Lecture Notes in Artificial Intelligence*, pages 279–294. Springer, 2007.
10. B. Möller, P. Höfner, and G. Struth. Quantales and temporal logics. In M. Johnson and V. Vene, editors, *Algebraic Methodology and Software Technology*, volume 4019 of *Lecture Notes in Computer Science*, pages 263–277. Springer, 2006.
11. T. Nipkow, L. C. Paulson, and M. Wenzel. *Isabelle/HOL — A Proof Assistant for Higher-Order Logic*, volume 2283 of *Lecture Notes in Computer Science*. Springer, 2002.
12. K. Rosenthal. Quantales and their applications. *Pitman Research Notes in Mathematics Series*, 234, 1990.
13. G. Sutcliffe and C. Suttner. The TPTP problem library: CNF release v1.2.1. *Journal of Automated Reasoning*, 21(2):177–203, 1998.