Elements of Mathematical Analysis in PVS *

Bruno Dutertre

Department of Computer Science,
Royal Holloway, University of London,
Egham, Surrey TW20 0EX, UK

Abstract. This paper presents the formalization of some elements of mathematical analysis using the PVS verification system. Our main motivation was to extend the existing PVS libraries and provide means of modelling and reasoning about hybrid systems. The paper focuses on several important aspects of PVS including recent extensions of the type system and discusses their merits and effectiveness. We conclude by a brief comparison with similar developments using other theorem provers.

1 Introduction

PVS is a specification and verification system whose ambition is to make formal proofs practical and applicable to large and complex problems. The system is based on a variant of higher order logic which includes complex typing mechanisms such as predicate subtypes or dependent types. It offers an expressive specification language coupled with a theorem prover designed for efficient interactive proof construction.

In previous work we have applied PVS to the requirements analysis of a substantially complex control system [2]. This was part of the SafeFM project which aims to promote the practical use of formal methods for high integrity systems. We used PVS to formalise the functional requirements of the SafeFM case study and to verify several safety critical properties.

The main problem we had with PVS was the limited number of pre-defined notions and pre-proved theorems; a non-negligible part of the work was spent in writing general purpose "background knowledge" theories. In general, we found that PVS provides only the most elementary notions and that some effort must be directed towards constructing re-usable libraries extending the pre-defined bases. This has been recognised by others and the new version of the system (PVS2 [15, 1]) comes with a largely expanded prelude of primitive theories and with better support for libraries.

Our experiment with the SafeFM case study showed that elements of mathematical analysis could be extremely useful for modelling hybrid systems. The case study is a control application including both discrete and analogue elements and the modelling involves continuous functions of time which represent physical variables. Reasoning about such variables can be considerably simplified if

---

* Work partially funded by EPSRC Grant No GR/H11471 under DTI Project No IED/1/9013
standard notions and results of analysis are available. This paper presents the development of a PVS library introducing such notions. The library defines convergence of sequences, limits of functions, continuity, and differentiation, and contains various lemmas and theorems for manipulating these notions.

Applications to hybrid systems were our prime motivation for developing such a library but integrating mathematical analysis to theorem proving can have other interests. Harrison [8] cites applications in areas such as floating point verification [9] or the combination of theorem provers and computer algebra systems [10].

The work presented in this paper is an example of use of PVS in a slightly uncommon domain, different from the traditional computer related applications. It was not obvious from the start whether PVS was a practical tool for doing "ordinary mathematics". Writing the library showed us that PVS could cope without much difficulty with the form of specifications and reasoning encountered in traditional mathematical analysis. In particular, the rich PVS type system was convenient for defining limits, continuity, and derivatives in a fairly natural way, very close to conventional mathematical practice. The library also makes use of some of the most recent features of PVS such as judgements and conversions.

All the proofs were performed using only the pre-defined set of proof commands, without any attempt to define new rules or proof strategies, the equivalents of HOL tactics and tacticals [7]. The high level commands available were powerful enough to handle automatically a large proportion of the proofs.

The remainder of this paper gives a brief introduction to PVS focusing on the aspects most relevant to the library development and presents the main components of the library. Section 4 discusses the qualities and limits of PVS for the application considered and gives a comparison with similar work.

2 An Overview of PVS

PVS is an environment for the construction and verification of formal specifications. The system provides an expressive specification language, a powerful interactive proof checker, and various other tools for managing and analysing specifications. PVS has been applied to large and complex examples in domains such as hardware [14], fault-tolerant protocols [15], or real-time systems [11].

The PVS logic is largely similar to classic higher order logic but with several extensions. The PVS type system is richer than Church's theory of simple types and supports subtyping and dependent types. PVS also includes mechanisms for writing parametric specifications. These features are essential and are described in greater detail in the following sections. We also outline the main characteristics of the PVS proof checker which influenced the formulation of certain aspects of the specifications. A more complete descriptions of the language and prover can be found in [1, 17, 18] and a more formal presentation of the PVS logic is available in [16].