Available online at www.sciencedirect.com
ScienceDirect
Future Generation Computer Systems 23 (2007) 721–726
FGCS FUTURE GENERATION COMPUTER SYSTEMS
www.elsevier.com/locate/fgcs

The Maple package SyNRAC and its application to robust control design☆

Hitoshi Yanami*, Hirokazu Anai

Information Technology Core Laboratories, Fujitsu Laboratories Ltd., Kamikodanaka 4-1-1, Nakahara-ku, Kawasaki 211-8588, Japan
CREST, Japan Science and Technology Agency, Kawaguchi Center Building, 4-1-8, Honcho, Kawaguchi 332-0012, Japan

Received 2 October 2006; accepted 17 October 2006
Available online 17 January 2007

Abstract

We have been developing a toolbox on Maple, called SyNRAC, for solving real algebraic constraints derived from various engineering problems. Its main tools are real quantifier elimination and the simplification of quantifier-free formulas. We illustrate algorithms implemented in SyNRAC, give some examples of how its commands are used, and present its application to design problems in systems and control theory.

Keywords: Robust control; Design; Symbolic computation; Quantifier elimination; SyNRAC

1. Introduction

The issue of robust control design has received increasing attention during the last few decades [1–3], mostly because of its outstanding applications to several domains [4,5]. In this field, the goal is to design controllers that optimize an objective function and that are robust or insensitive to parameter variations. To this purpose, it is required to find a set of real values for system parameters that minimizes a set of objective functions. A typical computational approach to solving this type of problem uses numerical methods. A fundamental way to do this is the following: set some real values for the parameters and compute the value of the objective function; pick another set of values and check again until a function value small enough to meet some performance requirement is found. There are many known algorithms that improve this naive method. If the problem satisfies a convexity condition, one can find the minimum value of the objective function with high efficiency. But such an algorithm cannot be applied in general to the nonconvex case. Another problem caused by numerical methods lies in the computational errors due to floating arithmetic. Numerical algorithms are very fast to compute, but inevitably contain errors caused by floating computation. To cover this defect there has been much work done, such as in interval arithmetic.

Our approach is based on another method that can circumvent the defects of the symbolic computation; we use symbolic computation with multi-precision integers, which frees us from the computational errors caused by rounding floating numbers. Symbolic computation methods have a shorter history than numerical ones, but nowadays they are widely applied to solving scientific and engineering problems [6]. This new tendency has resulted from the efficient symbolic algorithms introduced or improved during the last few decades, and by the advancement of computer technology that has hugely increased CPU power and memory capacity. In symbolic computation, every numeral is represented by a rational number and parameters are treated as indeterminates. The ring of multivariate polynomials over the rationals is the ground where we compute objects. Algebraic numbers are also within our scope; an algebraic number is defined as its minimal polynomial with an interval for root isolation.

The main method of our symbolic approach is *quantifier elimination*, which returns a quantifier-free equivalent for a given first-order formula of real closed fields, whose atomic formulas are polynomial equations, inequations, or inequalities. The optimization problem described above can be expressed in a first-order formula by introducing a slack variable. The

☆ Supported by CREST, Japan Science and Technology Agency.
* Corresponding author at: Information Technology Core Laboratories, Fujitsu Laboratories Ltd., Kamikodanaka 4-1-1, Nakahara-ku, Kawasaki 211-8588, Japan. Tel.: +81 44 754 2663; fax: +81 44 754 2664.
E-mail addresses: yanami@labs.fujitsu.com (H. Yanami), anai@jp.fujitsu.com (H. Anai).

0167-739X/$ - see front matter
doi:10.1016/j.future.2006.10.009

symbolic approach for this is characterized by its exactness, and with this approach the nonconvex case can be also dealt with.

We have been developing SyNRAC (an abbreviation for Symbolic–Numeric toolbox for Real Algebraic Constraints) on the Maple software for solving real algebraic constraints. SyNRAC is aimed at being a comprehensive toolbox composed of a collection of solvers for real algebraic constraints derived from various engineering problems. Using Maple/MATLAB as a platform has the following advantages: (1) Maple packages are automatically incorporated into MATLAB, which is widely used in engineering, via its "Extended Symbolic Math Toolbox", (2) They provide a good environment for realizing symbolic–numeric solvers (floating-point arithmetic, many numerical packages for, e.g., optimization).

So far, there is no quantifier elimination package on Maple, though Maple is one of the most widely used computer algebra systems. The focus of the implemented algorithms is on practically effective quantifier elimination (QE) for certain industrial/engineering problems and the simplification of quantifier-free formulas. Therefore SyNRAC provides a yet another implementation of quantifier elimination, but one which is still missing in Maple. The solvers to be addressed include several types of special QE procedures.

Currently the following algorithms are available in SyNRAC:

- special QE by the Sturm–Habicht sequence for sign definite conditions
- special QE by virtual substitution for linear formulas
- some simplification methods for quantifier-free formulas.

Furthermore, based on SyNRAC we have been developing some toolboxes tailored for specific application fields, e.g., robust control design, on MATLAB, which would be novel tools that provide new systematic design procedures for engineers.

We note that this work is strongly motivated by one of the authors' previous work concerning practically effective applications of QE to robust control design problems [7,8]. They showed that by integrating a reduction procedure for obtaining particular classes of formulas with a QE algorithm directed to such formulas, one can effectively solve practical control problems.

2. Special quantifier elimination methods

In general, a quantifier elimination algorithm, as is often the case with symbolic computation, has considerable computational complexity. It has doubly exponential behavior with respect to the number of quantifiers to be eliminated. One of the ways to circumvent this problem is to find a good class of formulas to which some specialized, more efficient QE algorithm is applicable. Such types of QE are called *special QEs*. In SyNRAC, there are two types of special QE algorithm that are sufficiently efficient for practical problems. These special QE algorithms as well as simplification methods for quantifier-free formulas are now available in SyNRAC. We discuss the QE algorithms in SyNRAC with their theoretical background in this section, and the simplification methods in the next.

2.1. Special QE for the sign definite condition

A formula of type $\forall x\ f(x) > 0$, where $f(x)$ is an integral polynomial over x satisfies what is called the *sign definite condition* (SDC). A special QE method based on the Sturm–Habicht sequence for this type of formula has been proposed by Gonzáles-Vega. This class of formulas is very significant because a quite wide range of important problems in robust control can be reduced to ones where the SDC holds [7].

2.2. Linear QE by virtual substitution

Another type of special QE procedure in our toolbox is quantifier elimination by so-called *virtual substitution* for linear formulas. A linear formula is one whose atomic subformulas are all linear with respect to the set of its quantified variables. Weispfenning first proposed such a QE algorithm for linear formulas. Loos and Weispfenning [9] have subsequently presented more efficient algorithms.

3. Simplification

When a quantifier is eliminated in a given first-order formula with a special QE procedure, its quantifier-free part usually gets larger. During a QE algorithm, formulas under manipulation tend to get extremely long, deeply nested and highly redundant, which sometimes even halts the procedure. That is why simplification procedures, which equivalently change a quantifier-free formula into a more concise one, are important in QE.

Simplification procedures implemented in SyNRAC are based on the methods in Dolzmann and Sturm [10]. Utilizing simplification algorithms combined with a special QE algorithm contributes to improve not only the efficiency of the computation but also the readability of the resulting formula.

Automatic formula simplifiers are implemented in REDLOG[1] and QEPCAD[2] (see [10] for possible simplifications). Several simplification rules including so-called order theoretical and additive smart simplifications are implemented in SyNRAC. These methods work quite well, especially when formulas are not deeply nested.

4. Example commands in SyNRAC

In this section we show some example commands to illustrate how commands in SyNRAC are used.[3]

After loading the SyNRAC package using the read command, you can use `qe_sdc` to solve an SDC compliant formula $\forall x > 0,\ f(x) > 0$. The first argument of `qe_sdc` is a polynomial f

[1] REDLOG is a QE package based on virtual substitution using REDUCE.

[2] QEPCAD is a general-QE package based on cylindrical algebraic decomposition.

[3] All computations were executed on a Pentium III 1 GHz processor.