Problem Solving and Theorem Proving with Symbolic Computation

A Special Topics Course Proposal

Prof. Marek Rychlik
Office: Math 605
phone: 621-6864
email: rychlik@math.arizona.edu
WWW: http://alamos.math.arizona.edu

October 17, 2003

Course Description

Goals

Symbolic Computation (also known as Computer Algebra) has found many applications as a complementary computational technique to the more traditional numerical computation. Effective use of symbolic computation involves sometimes sophisticated mathematical techniques and the knowledge of basic algorithms used in the implementation of the symbolic computation systems. These techniques are not covered in standard mathematics or computer science curricula. This course will provide an opportunity to learn them.

Most users of symbolic computation systems stick to high-level commands which completely solve the problems at hand. In this course we will learn more:

1. How to write new computer algebra systems, using tools like Lisp and Prolog and C++;
2. How to correctly design extensions to the existing systems, like Mathematica and Maxima.
3. How to automatically prove and discover mathematical theorems.

Topics covered

The material of this course will cover the following subjects:

1. Languages and data structures used in symbolic computation.
2. Samples algorithms of symbolic computation.
3. The structure of major symbolic computation systems.
4. Extended examples of calculations using symbolic computation systems from various mathematical disciplines.

We will include algorithms for manipulation of expressions, solution of algebraic equations, symbolic differentiation, automatic symbolic calculation of integrals, solution of ordinary and partial differential equations, perturbation and asymptotic expansions, Padé approximations.

We will also present various paradigms of symbolic programming: functional programming, rules programming, recursion, list processing and programming in logic.

The mathematical problems discussed will be mainly from the areas of ordinary and partial differential equations and computational algebraic geometry. Extended examples of applications of symbolic computation may include:

1. Perturbation expansion of the period of the van der Pohl equation, including an analysis of singularities in the complex domain.
2. Solving systems of algebraic equations resulting from various problems of mechanics, with an in-depth discussion of Gröbner bases.
3. Computing the Lindstedt series and perturbation series of Hamiltonian mechanics.
4. A Computer-assisted proof of the existence of the Feigenbaum fixed point.
5. Automatic calculation of symmetry groups of systems of non-linear ODE and PDE using Kuranishi-Cartan method.
6. Examples from control theory.
7. Examples from robotics.
8. Examples from signal processing.

Prerequisites

1. A senior-level or graduate course in linear algebra, vector calculus, and differential equations.
2. Plus at least one or a combination of the following
   (a) A working knowledge of a programming language.
   (b) Experience with a CAS like Mathematica, Maxima (formerly MACSYMA) or Maple.
   (c) Experience in the use of computers.