A System for Parallel Computer Algebra Programs

Stephen M. Watt[1]

Department of Computer Science
University of Waterloo
Waterloo, Ontario
Canada N2L 3G1

Small problems in computer algebra can be done with moderate time and storage. However, even simply stated problems of medium size can be very expensive to compute. Many such problems run out of storage or take excessive time. Large problems, if they can be done at all, can take hours. As in many other fields, we run into the question as to what is the most economical way to obtain raw processing power. We would like to investigate the practicality of using multiprocessing in computer algebra applications.

Many activities in symbolic mathematical computation are suitable for division into parallel processes. For example, when a sum is integrated, the first thing that is usually tried is to integrate each of the terms. At this high level, the tasks to be performed can be quite substantial so we expect the overhead of process management to be small by comparison.

We have constructed a prototype for running parallel computer algebra programs on a multiprocessor. The multiprocessor on which the system runs is a local area network of Vax 11/780's running Berkeley Unix version 4.2.[2] This program was written using the internet socket support provided by this version of Unix. The fact that the multiprocessor is a local area network is transparent to the user programs.

The prototype is a multiprocessing version of the Maple computer algebra system. It provides functions for dynamic process creation and interprocess communication. The communication primitives are used for synchronization.

The system consists of two parts: a program which implements the multiprocessing primitives and a small library of functions which communicate requests to that program. When running, a parallel program consists of a number of application processes, each with an associated system process

[1]Present address: Computer Algebra Group, Computing Technology Department, IBM Thomas J. Watson Research Center, Yorktown Heights, New York 10598 USA.

[2]Unix is a trademark of AT&T Bell Laboratories.

which acts as its agent. All input and output of an application process is filtered by its agent for embedded requests.

Having the multiprocessing facilities independent of the Maple kernel allows the application processes to be written in other languages. Thus the prototype provides a mechanism for inter-language communication which avoids the the overhead of repeated start-up of subordinate processes. For example, if one were to write a program that used both asymptotic analysis and Laplace transforms, then one could use Maple for the asymptotic analysis and Macsyma for the Laplace transforms. If a distributed Maple program had the need for an :q.inference server:eq., that portion of the code could be written in Prolog. To do this by installing corresponding multiprocessing facilities in each interpreter would be very difficult.

As an example, we will present a program for the computation of polynomial GCD's.