Parallelization of Quantifier Elimination on a Workstation Network *

Hoon Hong
Research Institute for Symbolic Computation
Johannes Kepler University
A-4040 Linz, Austria
e-mail: hhong@risc.uni-linz.ac.at

Abstract. This paper reports our effort to parallelize on a network of workstations the partial cylindrical algebraic decomposition based quantifier elimination algorithm over the reals, which was devised by Collins and improved by the author. We have parallelized the lifting phase of the algorithm, so that cylinders are constructed in parallel. An interesting feature is that the algorithm sometimes appears to produce super-linear speedups, due to speculative parallelism. Thus it suggests a possible further improvement of the sequential algorithm via simulating parallelism.

1 Introduction

This paper reports our effort to parallelize on a network of workstations a quantifier elimination algorithm for the first order theory of real closed fields. Since Tarski's first algorithm [24], various improvements and new algorithms have been devised and analyzed [5, 1, 21, 3, 9, 4, 11, 8, 22, 14, 15, 6, 19, 16, 18]. In spite of these significant improvements, however, the practical applicability of the methods is still limited due to the enormous computational time required. Thus the parallelization seems to be a natural step to take in bringing down the computation time.

We chose to parallelize the partial cylindrical algebraic decomposition based method [15], mainly because it is the only algorithm which has been fully implemented and in use, as far as we are aware. The algorithm is an improvement of Collins' method [5]. The improved algorithm has the same asymptotic time bound for the worst case (doubly exponential in the number of variables), but it is dramatically faster than the original for small inputs, which can be tackled using currently available machines.

There exist algorithms with better asymptotic complexity (both in sequential and parallel) [9, 22, 8, 11, 12, 13, 10]. However, these methods are not fully implemented yet, and also the analysis of [16] suggests that their computational

* This research was carried out in the framework of the Austrian science foundation (FWF) project S5302-PHY (Parallel Algebraic Computation) and the European project (ESPRIT II) POSSO (Polynomial Systems Solving).
requirement for small inputs might be quite big. As soon as the sequential implementation of these methods are available, it will be interesting to investigate the behavior of these algorithms on small inputs and also parallelize them.

In [23] Saunders, Lee, and Abdali report their work on parallelizing Collins’ original algorithm [5] on a shared memory machine. Their experiment show about 50% efficiency, due to the following two main bottlenecks: (1) A few tasks took very long, causing several other processors to be idle. (2) The lock operations on the shared heap caused heavy overhead. We tackled these two problems as follows:

- As mentioned earlier, we parallelized the author’s improved sequential algorithm [15] which, unlike the original algorithm, allows the cylinders constructed in almost arbitrary order, thus provides possibility for achieving better load balancing.
- We used a network of workstations where each processor accesses its own local memory and carries out garbage collection also independently of each other. Thus the overhead of lock operation does not exist. The communication overhead was made negligible by ensuring coarse granularity of tasks.

The improved sequential algorithm [15] is, in one respect, a “search” algorithm. Parallelization of such algorithms sometimes produces an interesting experimental experience of “super”-linear speedups, due to “speculative” parallelism. This phenomenon can be understood by considering the following simple scenario. Suppose that I am searching for a treasure, and I come to the beginning of two different paths. At the ends of both paths, a treasure lies, but I do not know the fact. Assume that it takes 10 seconds to follow one path and 1 second for the other. Now if I carry out a “sequential” search by checking one path after the other, the average search time is 5.5 seconds. But if I and my friend carry out a “parallel” search by checking both paths at once, then the average (also maximum) search time is 1 second. Thus I and my friend will experience a “super”-linear speedup of 5.5.

It is important to note that this advantage could also be had in an improved sequential version, one which simulates the parallel search through a kind of timesharing.

2 Overview of Sequential Algorithm

In this section, we give a high level description of the partial cylindrical algebraic decomposition based quantifier elimination algorithm [15]. Let $F^* = (Q_{f+1}x_{f+1})\cdots(Q_rx_r)F(x_1,\ldots,x_r)$ be a prenex formula of the first order theory of real closed fields where $Q_i$’s are quantifiers and $F$ is a quantifier free matrix. The algorithm, given such a formula, produces an equivalent quantifier-free formula $F'(x_1,\ldots,x_f)$. The algorithm proceeds in three steps: projection, truth invariant decomposition, and solution. We have parallelized only the second step, and thus we will go into some details of this step, while giving only brief descriptions of the other steps.