Lessons Learned when Comparing Shared Memory and Message Passing Codes on Three Modern Parallel Architectures

J. M. MacLaren and J. M. Bull

Centre for Novel Computing, Department of Computer Science,
University of Manchester, Oxford Road, Manchester M13 9PL, UK

email: {jon.maclaren,mark.bull}@cs.man.ac.uk

Abstract. A serial Fortran 77 micromagnetics code, which simulates the behaviour of thin-film media, was parallelised using both shared memory and message passing paradigms, and run on an SGI Challenge, a Cray T3D and an SGI Origin 2000. We report the observed performance of the code, noting some important effects due to cache behaviour. We also demonstrate how certain commonly-used presentation methods can disguise the true performance profile of a code.

1 Introduction

Micromagnetics is an area where simulation is of vital importance, enabling electronic engineers to model and predict the behaviour of magnetic materials. As in many other fields, accurate simulations are computationally demanding—parallel computers offer a means of meeting these demands. A serial micromagnetics code was parallelised in two ways, producing a Shared Memory code, and a Message Passing code. This paper describes this procedure, and compares the performance of the two codes on three different parallel architectures, and attempts to explain these results as fully as possible .

In Section 2, the subject code and its problem domain are described briefly. Details regarding relevant aspects of the target architectures are provided in Section 3. Methods for writing parallel Fortran on these machines are then chosen. Full details of how the parallel codes were implemented can be found in [7].

Results are presented for the codes in Section 4—these are found to be unusual for one platform in particular. The results are discussed in detail, with explanations being given for seemingly anomalous behaviour, and the deficiencies of some common performance presentation practices are exposed. Finally, in Section 5, we draw some conclusions.

2 Subject Code

The subject code simulates thin film media, such as hard-disk surfaces, which are made up from layers of many magnetisable grains. The code models the shapes,

sizes and locations of the grains, and uses this information to predict how the magnetic fields of the grains change in reaction to an applied, external magnetic field. Each grain has its own magnetic field, and so influences every other grain, making this an N-body problem. The interactions are solved by integrating N Landau Lifshitz equations[8] of the form

$$\frac{d\boldsymbol{M_i}}{dt} = \frac{\gamma}{1+\alpha^2}(\boldsymbol{M_i} \times \boldsymbol{H_T}) - \frac{\alpha\gamma}{1+\alpha^2}\frac{\boldsymbol{M_i} \times (\boldsymbol{M_i} \times \boldsymbol{H_T})}{|\boldsymbol{M_i}|}$$

for $i = 1, \ldots, N$, where N is the number of grains. Here $\boldsymbol{M_i}$ is the magnetic moment of the ith grain, α and γ are constants and the term $\boldsymbol{H_T}$ contains the contributions to the magnetic field from the externally applied field and from the influence of the other grains.

To evaluate the term $\boldsymbol{H_T}$ precisely for every grain would have complexity $O(N^2)$, so approximations are sought which yield acceptably accurate solutions in a reasonable time. Extensive work in the field of N-body problems has generated methods for constructing approximations of N particle systems which are of complexity $O(N \log N)$ or better [2, 5]. These methods group particles into a hierarchy of cells. An 'average' field is calculated for each cell, based on the fields of the particles contained within it, and these are used when calculating interactions with distant particles—the more distant the particle, the larger the size of cell which is used.

The subject code uses an algorithm based upon the Barnes-Hut method [2], but which is restricted to a 3-level (rather than n-level) hierarchy. The largest cells are used to govern the problem size and shape, e.g. 4×4 denotes a square problem area with 16 large cells (each of which are subdivided into 9 smaller cells, which in turn contain a total of 48 magnetisable grains). At each iteration of the code, the external field is incremented by a fixed amount and the system of ODEs is integrated to convergence, so that the magnetic fields reach an equilibrium. A variable order, variable step Adams method with error control is used for the integration. The code uses 400 of these iterations to show how the overall magnetic field of the particles follows a changing external field, giving a hysteresis loop. Further details of the subject code can be found in [9].

3 Target Machines and Programming Methods

There are many different kinds of parallel architecture in existence. Here, machines with different architectures are chosen in order to represent the currently popular classes of parallel machines. The three chosen architectures, SGI Challenge, Cray T3D, and SGI Origin 2000 are each described briefly. Methods for programming these machines are then selected.

The SGI Challenge is a true shared memory architecture, the main memory being interleaved among the processors, which are connected using a simple bus. Each processor has its own write-through level-one and level-two caches. An invalidate protocol is used to achieve cache-coherency, providing sequential consistency. The SGI Challenge machine which was made available had 512Mb