Examining Parallelisation of the Ising Model

Department of Physics, University of Bristol.
(Dated: February 9, 2017)

Methods of capturing data concerning the Ising model are parallelised using OpenMP and MPI. The effect of lattice size is examined for its effect on the transition temperature and the methods of parallelisation are examined for their speed up and efficiency.

1. INTRODUCTION

Good Monte Carlo techniques are, by their nature, "not cheap" [1]. A fast program is therefore akin to a program that can obtain good data from a Monte Carlo technique, and an effective method of achieving such a program is by making use of parallel architectures.

1.1 Parallel Processing

A process is parallelised with the intention of improving the speed of a program. Often however, an improvement is partially or even completely drowned in the overheads that are implicitly produced when writing it [2]. In OpenMP these overheads come predominantly as a result of splitting into threads and scheduling the chunks of work assigned to each thread. The speedup is also limited by factors such as Amdahl's law, idle threads and the requirement for cache-coherency, as is the case for shared memory parallel programming such as OpenMP [3].

By contrast, MPI has distributed memory, so the overheads are dominated by message the passing between threads necessary for the exchange of data between processors [4]. This overhead is a result of both latency and bandwidth, where latency is the cost of initiating the message and bandwidth is the sustained data rate once a message has begun. The easiest and most effective factor to reduce is latency is simply by reducing the occurrence of message passing to a minimum.

With these overheads in mind, this paper will examine the speedup obtained in parallelising the Ising model by OpenMP and MPI where speedup is defined as

$$S_p = \frac{T_1}{T_p} \tag{1}$$

where S_p is the speedup for p processors and T_p is the time taken for the examined section of code to take with p processors implemented.

In general a computer user is most interested in speedup for a process but a better quantity for the programmer is the efficiency, defined as

$$E_p = \frac{S_p}{p}. \tag{2}$$

This quantity will give insight into the effectiveness of the parallelisation of the program.

1.2 The Ising Model

Magnetisation arises as a result of a system's competing desires to both maximise entropy and minimise energy [5]. For a given material, the mediating factor between the two is temperature. If temperature is large then disorder dominates and magnetisation falls to zero, if temperature is small then a preferred arrangement can cause a material to exhibit a finite magnetisation. Between these circumstances exists a transition temperature [6] at which certain materials go from permanent magnetism to magnetism induced only by an external magnetic field. This behaviour and resultant features are well examined by the Ising model.

Proposed by Lenz and solved by Ising in 1D (subsequently following with an erroneous conclusion) [7], the Ising model approximates a material as a lattice of spin components, each of which is either in the state spin-up $(+1)$ or spin-down (-1). The interaction between lattice points is by nearest neighbours only and is equal to $-J$ for parallel spins and $+J$ for anti-parallel spins, where J is an energy interpreted as a quantum exchange parameter, dictating the type of material considered. $J > 0$ (ferromagnetic) means the spin-spin interaction acts to make neighbouring spins the same the same and so it would, in general, align with an external field. $J < 0$ (anti-ferromagnetic) means the spins want to anti-align and so ideally half would would follow an external field and half would not.

Following this model, the magnetisation, M, and the total energy, E, of a configuration α are simply defined as

$$M(\alpha) = \sum_{i}^{N} s_i \tag{3}$$

and

$$E(\alpha) = -J \sum_{i,j=\mathrm{nn}(i)}^{N} s_i s_j - H \sum_{i}^{N} s_i \tag{4}$$

where N is the number lattice sites/spins, s_i=± 1 is the ith spin and H is an external magnetic field.

From these two quantities, the transition temperature can be considered by examining discontinuities in their derivatives with respect to temperature. In order to determine them for a specified temperature however, we require their expected values at that temperature.

The probability of a configuration α is given by the Boltzmann distribution

$$\mu(\alpha) = \frac{1}{Z}\exp(-\frac{E(\alpha)}{k_B T}) \tag{5}$$

where k_B is the Boltzmann constant, T is the temperature of the system and Z is the partition function of the system.

This distribution is the mediation between minimised energy and maximised entropy. For a lattice of N sites, there are 2^N configurations available with probabilities given by Eq. (5). To generate a configuration, we sample from this distribution such that the expected value of an observable A, given by the sample mean of M samples,

$$< A >= \frac{1}{M}\sum_{k=1}^{M} A(\alpha_k), \tag{6}$$

converges rapidly to the true mean value. The problem then becomes generating a weighted set of configurations.

2. METHODS

2.1 The Metropolis Algorithm

A symmetric Markov chain, composed of the available configurations of the system and converging to Eq. (5), can be simply constructed for the Ising model. If one spin is inverted, the resultant change in energy, ΔE, is easily evaluated due to two factors. Firstly, only $1 + 2D$ terms in Eq. (4) change, where D is the dimensionality of the system, and secondly each term changes by a factor of -1.

The transition probability of α to β for the Markov chain is $p_{\alpha,\beta}$ such that

$$\mu(\alpha) = \sum_{\beta} \mu(\beta) p_{\alpha,\beta}, \tag{7}$$

where the sum considers all configurations varying from α by an inversion of one spin. Therefore, if we consider one lattice site, then we can write

$$p_{\alpha,\beta} = \frac{\mu(\alpha)}{\mu(\beta)} = \exp(-\frac{E(\alpha) - E(\beta)}{k_B T}) = \exp(-\frac{\Delta E}{k_B T}) \tag{8}$$

for $\Delta E > 0$, and

$$p_{\alpha,\beta} = 1 \tag{9}$$

for $\Delta E < 0$. When uniformally picking a site on the lattice to consider each time, this probability can be used to move the lattice towards equilibrium via an appropriate path through phase space.

Repeatedly following this, sequentially picking a new site from the new configuration, is the Metroplolis algorithm. It can be summarised as follows: 1) Pick a random site on the lattice. 2) If the inversion of this spin lowers the total energy of the lattice, then accept the change. 3) If the inversion increases the energy, then accept the change with probability p. 4) Repeat.

The Metropolis algorithm, of which one iteration is defined as completed after N random sites are picked and examined for a lattice composed of N sites, was employed in obtaining all data considered in this paper. To evaluate the expected values of the observables of interest the lattice is initialised to a random configuration, and K iterations of the algorithm are performed upon it before collecting any data. This is the burn-in time required to remove artifacts from a random initialisation and to move the lattice into equilibrium with the temperature of interest. Data is then taken every iteration for a large number of iterations and the sample average is hence determined.

2.2 OpenMP

Embarrassingly parallel techniques were employed to capture data using OpenMP. For these cases each thread has their own lattice and performs the Metropolis algorithm on it without interaction, hence determining their observables independently. The results are subsequently combined to either increase the number of samples for a data point, or, when considering the effect of lattice size where each thread has a differently sized lattice, to collect more data points. As a result the Metropolis algorithm is not parallelised by OpenMP since reliable data is more quickly obtained by parallelising the loop in which the algorithm is applied.

2.3 MPI

Two methods were employed when examining parallelisation by MPI, varying by the technique in attempting to reduce to quantity of message passing. Both methods involved the workers having a (mostly) equal fraction of the grid and the adjacent slice of the two neighbouring fractions. The lattice is split up along one axis, the x axis, which is generally not ideal since the ratio of message passing to computational work is proportional to the surface area to volume ratio of the fractions. However, this drastically simplifies the program and, for one method, can be argued to be beneficial to the speed.

Messages containing information on the spins of the neighbouring fractions are passed when required by the Metropolis algorithm to update a site.

The first of the two methods, henceforth "the probe method", involves use of the function MPI_Iprobe to ask whether a message is waiting to be received. Should a message be available, then it is subsequently received. An issue with this method arises from the computational cost of this function; it is called twice for every iteration of the Metropolis algorithm so if it is expensive then the speed of the method is consequently very poor. It is for this reason that splitting the lattice along one axis may be beneficial for this method. For a lattice split along one axis, only two calls of MPI_Iprobe are required whereas, if it were split along two axes it would require four calls, and it would require six calls if it were split along three axes.

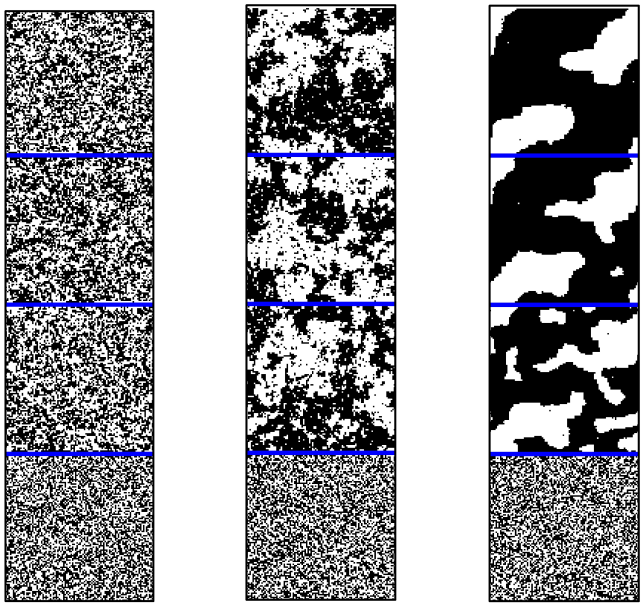
For the second method, "the mirror method", each thread picks the same point on the lattice. This avoids any conflict in selecting neighbouring sites and means all message passing occurs at the same time. In addition, the threads keep a list of whether the neighbouring sites have been selected and a message is passed only if the required spin is possibly different to the recorded value. Then all unknown spins concerning the relevant boundary are updated from a single message. This method has minimal communication at the cost of more processing and it is expected to be greatly beneficial for the speed-up. An obvious problem in this method is the requirement for each thread to pick the same random number. Although the sites selected on the lattice are not strictly random, this appears to not introduce any bias in the observables at equilibrium nor in the dynamical behaviour of the lattice as it approaches equilibrium. This is anticipated to be the case for all lattices that are not very small.

A notable difference between the two methods arises in their generality. The probe method works for any grid size for any number of threads whereas the mirror method works only if the x dimension is divisible by the number of threads. Therefore both methods are potentially useful in parallelising the Metropolis algorithm.

3. RESULTS

3.1 Physical

Figure 1 shows, amongst other things, how the lattice varies with temperature. A measure of this variation is the correlation length which is, in turn, a measure of the distance over which spins respond to other spins. When at low T, the correlation length is small since, although the spins tend to be aligned with each other, flipping one spin will hardly affect another (unless they are neighbours). When at high T, the correlation length is again low since the spins fluctuate rapidly and do so nearly independently of others. However, when close to the transition temperature, there are persistent domains of parallel spins despite the spins constantly changing. Therefore



(a) Above criticality: $k_BT = 5.$

(b) Close to criticality: $k_BT = 2.3.$

(c) Below criticality: $k_BT = 1.$

FIG. 1: Images through time for three temperatures of a ferromagnetic 100x100x1 lattice in the absence of an external magnetic field. Images were taken sequentially after 50 iterations of the Metropolis algorithm from an initially randomised state. A black pixel represents a spin-up lattice site, a white pixel represents a spin-down lattice site and the blue line indicates a step in time with time increasing up the figure.

greater close range order exists, the spins are far more affected by each other and hence the correlation length is large.

The correlation function is determined from

$$< s_i s_j > - < s_i >< s_j > \tag{10}$$

and exhibits the form

$$g(R) \sim \frac{1}{R^\eta} \exp(-\frac{R}{\xi}) \tag{11}$$

where R is the distance between considered lattice sites, η is a critical exponent and ξ is the correlation length.

In all cases shown in Fig. 2, correlations are strong near the origin indicating that a spin's influence is strongest on its nearest neighbours. The correlation then decays as R increases according to the correlation length. The decay constant, or correlation length, is greatest when close to the transition temperature indicating, as expected, greater short range order close to the transition temperature. The correlation does not in general tend to zero, but instead tends to $< M >^2$, an indication

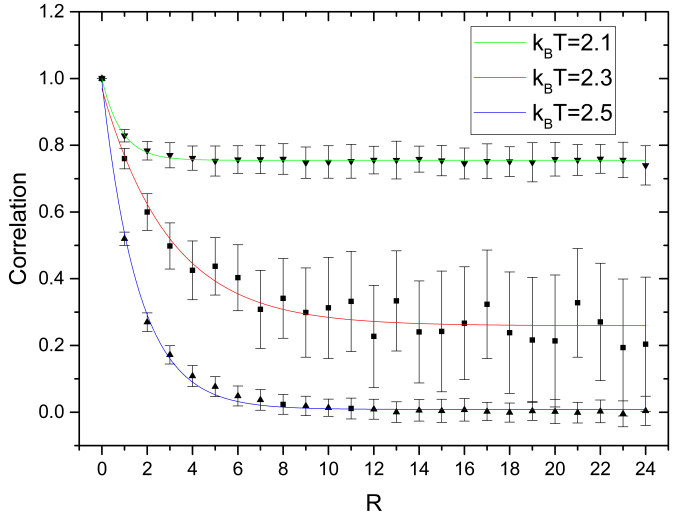


FIG. 2: Spin correlation functions for a ferromagnetic 50x50x1 lattice above ($k_BT = 2.6$), below ($k_BT = 2.0$) and close to ($k_BT = 2.3$) the transition temperature. The determined correlation lengths are 0.85 ± 0.02 for $k_BT = 2.1$, 3.0 ± 0.3 for $k_BT = 2.3$ and 1.61 ± 0.04 for $k_BT = 2.5$.

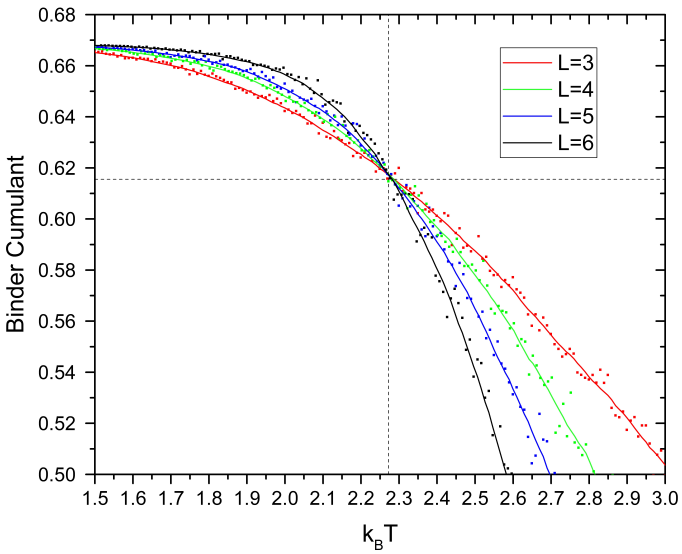


FIG. 3: Binder cumulant vs k_BT for various square lattices with sides of length L. The dashed line indicates the coordinate of the intersection of the four coloured lines: $k_BT = 2.27 \pm 0.01$, $U = 0.616 \pm 0.002$.

of the long range order of the system. Therefore, from this figure, we can find a definition of the long rage order vs short rage order of the system as well as a definition for the critical temperature of the system.

A second interesting effect arising from a computational Ising model is how the lattice size affects the transition temperature. This occurs because when the correlation length is of the order of the lattice size, then the typical size of a cluster of parallel spins is the size of the lattice. Hence the transition temperature is reached. When considering an infinite lattice, then the correlation length must diverge at the true transition temperature, so to go to the size of the lattice.

A way to determine the critical temperature from finite lattices is by means of the fourth-order Binder cumulant, U, which is defined as

$$U = 1 - \frac{< M^4 >}{3 < M^2 >^2}. \qquad (12)$$

At the critical point, U goes to a critical and, crucially, nearly universal value. Therefore, for all lattice sizes, U will tend to the roughly same value as T tends to T_c; providing a useful way to estimate the critical temperature when only finitely sized lattices are accessible. From Fig. (3) shows the intersection point for 2D lattices, giving a critical temperature estimate of $k_BT_c = 2.27 \pm 0.01$ and Fig (4) shows the intersection point for 3D lattices, giving a critical temperature estimate of $k_BT_c = 2.27 \pm 0.01$.

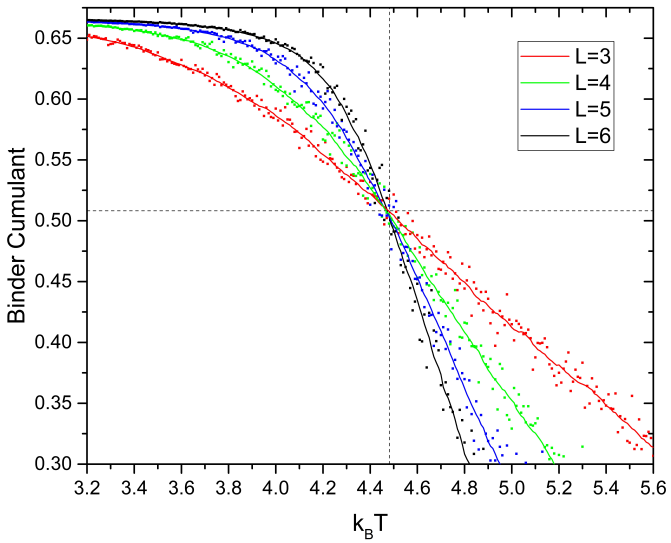


FIG. 4: Binder cumulant vs k_BT for various cube lattices with sides of length L. The dashed line indicates the coordinate of the intersection of the four coloured lines: $k_BT = 4.8 \pm 0.2$, $U = 0.51 \pm 0.1$.

3.2 OpenMP

Methods of parallelising with OpenMP, as discussed, were varied according to the required data. The following examined speedup and efficiency concerns a simple, embarrassingly parallelised, iteration loop for the Metropolis algorithm. The loop involves 96 iterations of the algorithm on a $40 \times 40 \times 40$ sized lattice.

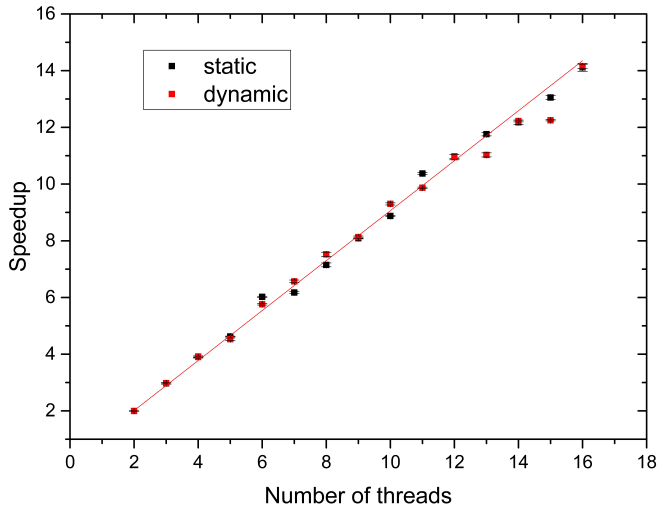


FIG. 5: Dependence of speedup on the number of threads for a $40 \times 40 \times 40$ sized lattice. The gradient of the line displayed is 0.88 ± 0.01.

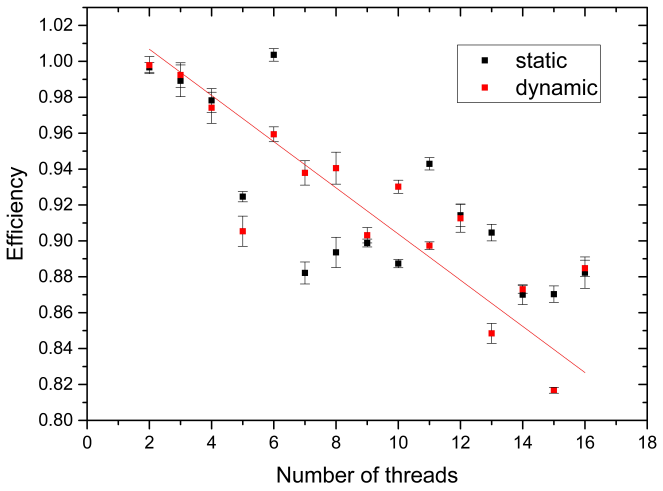


FIG. 6: Dependence of speedup on the number of threads for a $40 \times 40 \times 40$ sized lattice. The gradient of the line is -0.013 ± 0.002.

Figure 5 shows a linear dependence of speedup on the number of threads, an indication of an embarrassingly parallel process. The speedup is not perfect however, as the gradient is less than 1.

Figure 6 also shows a linear dependence with small negative gradient on the efficiency indicating that the linear dependence in fact does tail off given enough threads in use.

3.3 MPI

Analysis of the speed up when using MPI is contained to examination of the mirror method. The probe method was deduced to be consistently evasive of any acquired speedup; the MPI_Iprobe function is too intensive. The following data was obtained with the intention of determining the general dependence of efficiency against number of threads for large lattices, where the computation time dominates the communication. This is then compared to the shape of the dependence for a small lattice, where communication is a far more influential factor.

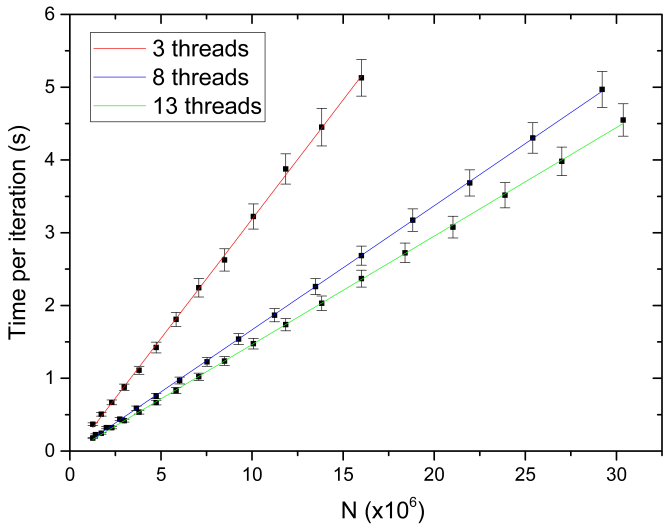


FIG. 7: Displaying the linear relation between time per iteration and number of sites for large cubic lattices. The lines shown are for three threads, with a gradient of $(3.22 \pm 0.02) \times 10^{-7}$, eight threads, in with a gradient of $(1.695 \pm 0.005) \times 10^{-7}$, and thirteen threads, with a gradient of $(1.484 \pm 0.004) \times 10^{-7}$.

The time taken per iteration for a general "large" lattice was determined by considering the time per iteration for a number of large cubic lattices. A graph of time per iteration against number of lattice elements was produced for one through sixteen threads. From this, the gradient of the linear relation between the two (shown for a few different numbers of threads in Fig. 7) was used to determine a general time per lattice site.

As a result, the dependence of speedup and efficiency on the number of threads were determined through Eq. 2.

The efficiency was determined to decay to zero according to a decay constant of -0.217 ± 0.002 for large lattices and -0.203 ± 0.007 for the $24 \times 24 \times 24$ lattice. The notable difference between the two lies in the initial amplitude of this decay which deviates by 0.101 ± 0.007.

4. DISCUSSION

4.1 OpenMP

OpenMP was parallelised with good efficiency, the methods of doing so were very simple but effective. Of note in the

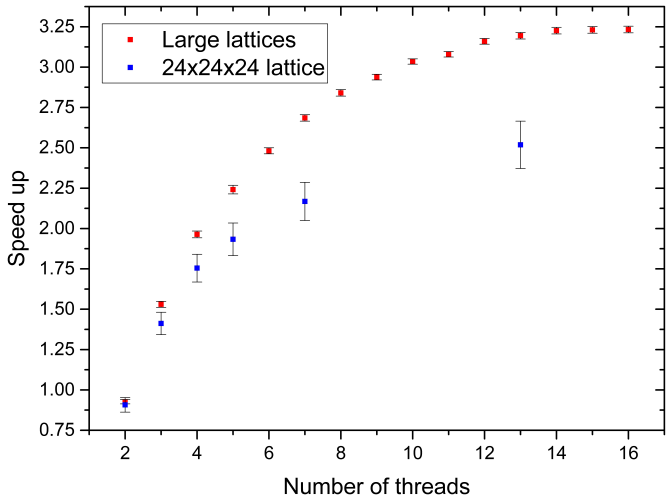


FIG. 8: The dependence of speedup on the number of lattice sites for the mirror method for large lattices and a small $24 \times 24 \times 24$ lattice.

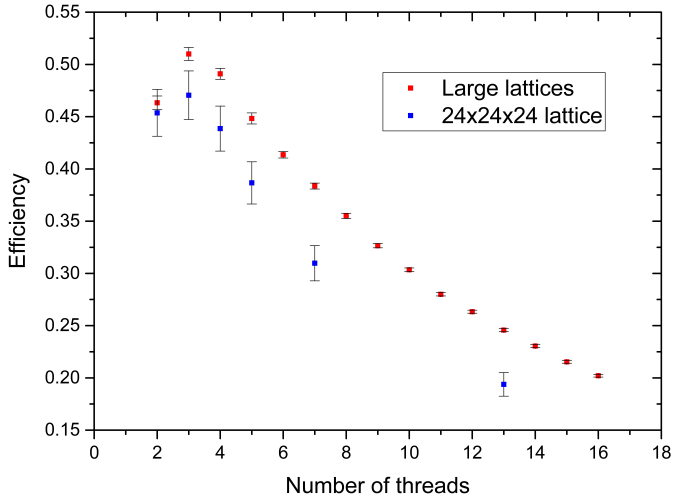


FIG. 9: The dependence of efficiency on the number of lattice sites for the mirror method for large lattices and a small $24 \times 24 \times 24$ lattice

results is that the gradient is in fact less than 1 and the efficiency still falls with increasing number of threads. This is due in part to the required set up and take down before and after performing the algorithm which cannot be parallelised and thus reduces the maximum speedup in accordance with Amdahl's law, but also to the scheduling of the chunks of work to each thread. This would obviously take slightly longer with more threads.

The difference between dynamic and static scheduling was not evident in either figures but was expected to show the factors of the number of iterations. For a number of threads that is a factor of the number of iterations which is followed by a number which is not a factor, the speed up between this increment in threads is slightly less than otherwise expected. This is because the maximum number of iterations performed by a thread remains the same. The reason this was not evident is because, since there are less threads performing the maximum number of iterations, the expected time taken by the slowest, limiting, thread is lower.

4.2 MPI

Parallelisation of the Ising model by MPI requires message passing across the boundaries of adjacent lattice fractions. This only ever occurs when a surface element of a fraction is selected, therefore it is the ratio of surface-area to volume of each fraction that dictates how large the speedup will be. Surprisingly, even moving to fractions with all elements on the boundary, speed up was still observed. This is probably due to the effort to reduce communications in the mirror method which causes messages to be passed only when absolutely necessary. However, the speed up is greatly diminished and the efficiency also consequently drops.

Notably on Fig. 9 the efficiency of two threads is less than the efficiency of three. This is a result of the drop in performance that occurs when two threads are used as opposed to one. This drop occurs because with one thread there is no requirement for sending, splitting, receiving and trading information between threads. However, with two threads, the lattice is both sent and received and information is passed from one thread to itself when a boundary element is picked; obviously wasted time that is apparent in Figures 8 and 9.

5. CONCLUSIONS

OpenMP and MPI are both examined for the dependence of speedup and efficiency displaying the embarrassingly parallel nature of the program for OpenMP and the effectiveness of the parallelisation for MPI. Using OpenMP is quicker at obtaining data since this Monte Carlo method requires a large number of repeats for each observable to be close to the expected value. Using MPI was, however, far more interesting from a programming perspective and highlighted the requirements for an optimally parallelised program.

The Ising model was examined at criticality, showing that the correlation length affects the transition temperature and determining that the transition temperature is $k_BT = 2.27 \pm 0.01, U = 0.616 \pm 0.002$ for a 2D lattice and $k_BT = 4.8 \pm 0.2$, $U = 0.51 \pm 0.1$ for a 3D lattice. Both results for the critical temperature compare well with accepted values.

[1] Binder, K. (1979). Monte Carlo methods in statistical physics. 1st ed. Berlin: Springer-Verlag.
[2] Pacheco, P. (2011). An introduction to parallel programming. 1st ed. Amsterdam: Morgan Kaufmann.

[3] Chandra, R. (2001). Parallel programming in OpenMP. 1st ed. San Francisco, Calif.: Morgan Kaufmann Publishers.
[4] Pacheco, P. (1997). Parallel programming with MPI. 1st ed. San Francisco, Calif.: Morgan Kaufmann Publishers.
[5] Simon, B. and Sokal, A. (1981). Rigorous entropy-energy arguments. Journal of Statistical Physics, 25(4), pp.679-694.
[6] Lee, T. and Yang, C. (1952). Statistical Theory of Equations of State and Phase Transitions. II. Lattice Gas and Ising Model. Physical Review, 87(3), pp.410-419.
[7] Brush, S. (1967). History of the Lenz-Ising Model. Reviews of Modern Physics, 39(4), pp.883-893.