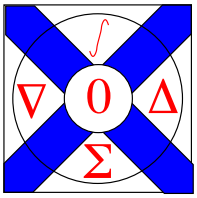

Computational Science Technical Note **CSTN-205**

Optimising Computations for Evaluating Ising and Potts Model Partition Functions by Exact Enumeration

K. A. Hawick and D. P. Playne

2013

The Ising and Potts discrete lattice are useful baselines of comparison for many systems and theoretical calculations in statistical physics. While these models are traditionally studied using Monte Carlo sampling techniques it is also useful to exactly enumerate their partition functions using brute-force coding techniques. Recent advances in bitwise manipulation and parallel processing technology have made these techniques computationally feasible for the Q=3,4 state Potts model as well as the 2 state Ising spin model. We report on bit-packing and graphical processing unit implementations to improve the number of model states that can be exactly enumerated per second and discuss implications for uses of this approach with various observables.

Keywords: partition function; enumeration; brute force; summation; multicore, GPU

BiBTeX reference:

```
@INPROCEEDINGS{CSTN-205,
        author = {K. A. Hawick and D. P. Playne},
        title = {Optimising Computations for Evaluating Ising and Potts Model Partition
                Functions by Exact Enumeration},
        booktitle = {Proc. 9th International Conference on Foundations of Computer Science
                (FCS'13)},
        year = {2013},
        number = {CSTN-205},
        pages = {FCS4072},
        address = {Las Vegas, USA},
        month = {22-15 July},
        publisher = {WorldComp},
        institution = {Computer Science, Massey University,m Auckland, New Zealand},
        keywords = {partition function; enumeration; brute force; summation; multicore,
                GPU},
        owner = {kahawick},
        timestamp = {2013.04.22}
}
```


Prof Ken Hawick, Computer Science, Massey University, Albany, North Shore 102-904, Auckland, New Zealand.

Technical Report CSTN-205

Optimising Computations for Evaluating Ising and Potts Model Partition Functions by Exact Enumeration

K.A. Hawick and D.P. Playne
Computer Science, Massey University, North Shore 102-904, Auckland, New Zealand
email: {k.a.hawick, d.p.playne }@massey.ac.nz
Tel: +64 9 414 0800 Fax: +64 9 441 8181

April 2013

ABSTRACT

The Ising and Potts discrete lattice are useful baselines of comparison for many systems and theoretical calculations in statistical physics. While these models are traditionally studied using Monte Carlo sampling techniques it is also useful to exactly enumerate their partition functions using brute-force coding techniques. Recent advances in bitwise manipulation and parallel processing technology have made these techniques computationally feasible for the Q=3,4 state Potts model as well as the 2 state Ising spin model. We report on bit-packing and graphical processing unit implementations to improve the number of model states that can be exactly enumerated per second and discuss implications for uses of this approach with various observables.

KEY WORDS
partition function; enumeration; brute force; summation; multicore, GPU.

1 Introduction

The Ising model [17] is a well studied statistical mechanics model of a magnetic system exhibiting a phase transition [33]. The Ising model has been very widely used as a baseline of comparison for many real systems, theoretical [21,34] and simulation models [16] in statistical and solid state physics [31], and in other areas such as neural models of the brain and complex networks such as polymer systems [22]. The Potts model [30] extends the Ising system by allowing more than just two simple spin states. The Ising and Potts models are both formulated as a discrete and regular lattice of individual spins that can take on a number of different values but which will align (ferromagnetic model) or anti-align (anti ferromagnetic model) with their nearest neighbouring sites. The degree of alignment changes drastically at a critical temperature - known as the Curie temperature for real systems [38]. Above T_c the spins are still random with no large spatial structure, whereas below T_c large droplets and regions of aligned spins spontaneously form.

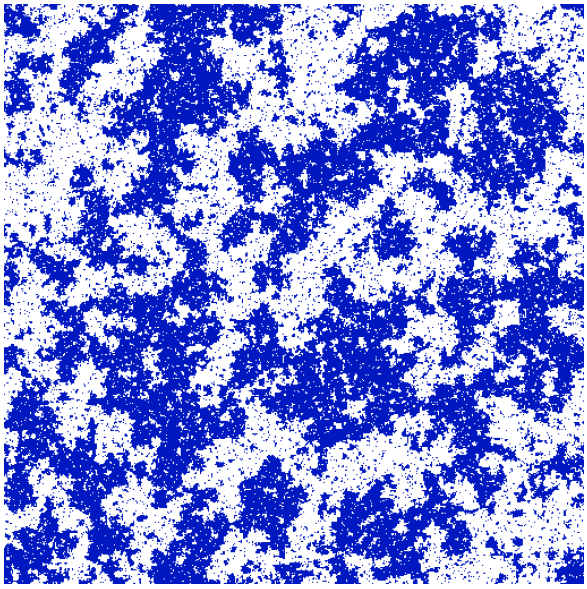

Figure 1: Long range clusters in a Critical 512^2 Ising model.

Figure 1 shows a sample Ising model system around its critical temperature. These models are usually studied using Monte Carlo sampling techniques [6, 7, 15, 25]. A model system is initialised at a high temperature with uniformly random spins and then quenched to a particular temperature T of interest and an algorithm like those of Metropolis or Glauber is used to allow the system to relax or equilibrate at that temperature. Measurements can be made on the simulated system and if the transition probabilities for the changes are chosen correctly according to a Boltzmann energy factor the procedure samples the various discrete starts according to their importance or probability of occurrence.

It is useful however to attempt to evaluate the partition function [11, 20, 32] – or sum over the individual states – of these models directly. The number of states M_2 of the 2 state Ising

model grows as $m = 2^N$ where $N = L^d$ is the number of spin sites. M grows very fast indeed and the system shown in Figure 1 has $N = 512^2$ spin sites and so has $2^2 62144$ different discrete states in its configuration space. The number of states in a Q-state Potts model system grows even faster – as Q^N. It is not feasible to enumerate the states explicitly for such a large system, but nevertheless it is valuable to explore changes to the partition function by studying small systems.

A great deal of work was done in the 20th Century in attempting to evaluate Ising model partition functions [13] for small systems [19] and has covered various model sizes [5] and special cases to try to understand for example finite size effects [2]. The Ising model has been solved analytically for the 2 dimensional case although it is still instructive to compare its known properties with those that result from enumerating its partition function on small square lattice Ising systems [23]. The 3 dimensional system still poses a challenge and its properties have only been approximately evaluated using simulation techniques. It is therefore of great and continued interest to understand the properties of a 3 dimensional Ising system using even a partial enumeration of its partition function [3, 4, 29].

There is also continued interest in exactly enumerating other systems such as the Potts model either directly [10, 24] or using symmetries or other approaches to restrict the state space traversed [9, 36].

Recently however advances in parallel processing technologies have allowed us to enumerate the states in small Potts systems. Our focus in this present paper is to investigate the computational feasibility of exact enumerations of 2 and 3 dimensional Ising and Potts systems using ordinary programming techniques; bit packing to keep the data structures in memory; and data parallel techniques using graphical processing units (GPUs). A decade of advances in processing power means that what were month long calculations in the 1990s are now feasible in a matter of hours. We obtain the exact energy spectrum and its populations from evaluating the partition function [1] but these new techniques open up scope for tracking more elaborate observables than just the energy as part of the traversal of the model's state space.

We focus on the zero field Ising [37] and Potts system but finite field versions of the models also offer scope for further work. Other related problems such as the inverse Ising model [27] involving deduction of the temperature from observed state samples, may also benefit from improved partition enumeration information. We believe that enumerations of the individual component size histograms may give insights into droplet formation and noise and fluctuations [12] in the model.

Our article is structured as follows: In Section 2 we briefly summarise the Ising and Potts models before giving a description of the statistical mechanical partition functions in Section 3. We describe our implementation methods using bit packing to keep data structures in memory cache and graphical processing unit data parallelism in Section 4. We present some timing and performance results and some preliminary partition function spectra for the Potts system in Section 5. We discuss the computational scope for this technique in Section 6 and offer some tentative conclusions and ideas for further study in Section 7.

2 Ising & Potts Models

The Ising model is formulated as a set of spin variables $\sigma_i = \pm 1$ arranged on a lattice - usually on a d-dimensional grid of size $N = L^d$ where length L is as large as can practically be simulated.

The energy functional or Hamiltonian of the Ising model is:

$$\mathcal{H} = -\mathcal{J} \sum_{\langle i,j \rangle} \sigma_i \sigma_j \tag{1}$$

This has no explicit time dependence or dynamical scheme associated with it, and so one must be imposed artificially. We do not give details of the normal Monte Carlo dynamical sampling approach here as it is well described in the literature [7, 8].

The main point to emphasise for the work reported in this present paper is that each spin is independent and in the case of the Ising model can take on $Q = 2$ states. A small Ising system of for example $N = 4^2, 5^2$ still has a considerable amount of symmetry present in it. There are a finite number of possible energy levels and they have a large degree of degeneracy - that is a great many of the possible arrangements of the spin variables in the system leads to the same energy level.

The Potts model extends the Ising model by allowing the "spins" to take on a discrete range of integer values. The consequence is that for a Q state system the number of possible arrangements of spin values goes as a power of Q which obviously grows even faster for high Q.

3 Partition Function

The thermodynamic partition function or sum over the states of the system is usually denoted as Z and is given by:

$$Z_N(T) = \sum_{\sigma_1} \sum_{\sigma_2} \cdots \sum_{\sigma_N} \exp\left(-\mathcal{H}(\{\sigma_i\})/k_B T\right) \tag{2}$$

where each degree of freedom σ_i takes the 2 values 0,1 for the Ising model or the Q values for the Q-state Potts extension of the Ising model. There are in fact a great deal of configurations that have identical energies due to symmetries and so the partition function can be expressed as:

$$Z_N(T) = \sum_l g_l \exp\left(-E_l/k_B T\right) \tag{3}$$

Where the sum is over all the energy levels labelled by l, and each level has degeneracy g_l.

The exact pattern of degeneracies varies somewhat for the small systems we are able to enumerate depending upon whether they are even or odd in their lengths.

Since our system is discrete we can evaluate them by building up an exact histogram of how many states fall into each energy level. This exact histogram can then be used to evaluate other statistical and theoretical properties of the model. Once the partition function [14] or an approximation to it is obtained there are other techniques such as locating its zeros [18] that can be applied to make further deductions on the model properties. In this present paper we only evaluate the energy level degeneracies and use this as a means of scoping the computational effort and attainable performance to make further studies.

This is a powerful technique but its use is limited by the rapid growth of the number of states $M_S = Q^{N=L^d}$ in the model. In practice we have been limited by computer clock speeds and memory performance to only very small systems sizes. We can however explore various computational optimization techniques to speed up this calculation so that new Q values and higher system sizes might be explored.

4 Enumeration of States

The easiest method of enumerating through all the possible combination of spin variables σ_i is to create an array of integer values to represent the system. The possible system states can be generated by iterating through every combination of values in this array. For an Ising model each site can be $\{0,1\}$ while for a Potts model each site can have the values $\{0..Q\text{-}1\}$.

Listing 1: Array enumeration algorithm uses an array `spin` to store all of the system states and a series of `for` loops to iterate through the possible states. The array `n` stores the indexes of each site's neighbours and NORTH and WEST are used as indexes into this array.

```
int spin[N];
for(spin[0]=0;spin[0]<Q;spin[0]++)
..
for(spin[N-1]=0;spin[N-1]<Q;spin[N-1]++) {
  long long E = 0;
  for(int i = 0; i < N; i++) {
    E+=(spin[i]==spin[n[i][NORTH]])?-1:1;
    E+=(spin[i]==spin[n[i][WEST] ])?-1:1;
  }
  histogram[E+offset]++;
}
```

The energy of a system can be calculated from the number of like-like bonds in the system. To determine the number of like-like bonds in a system, all the sites in the array must be iterated through and compared to the neighbouring values. This is slow for practical reasons however and if we can get all the neighbours and spin state information in cache it will be considerably faster to traverse than if main memory accesses are needed. An example of the code to iterate through every possible state for a system stored in an array and calculate the energy of each system can be seen in Listing 1.

4.1 Bit Enumeration

An Ising system is a series of spins that are either up or down, and so each site can be represented by a single bit. For an Ising system that is small enough to be enumerated, these bits can be stored in a single 64-bit integer. A 64-bit integer can represent an Ising system up to a maximum size of 8x8 or 4x4x4. The process of generating the different Ising configurations is then simplified to iterating through the possible integer values $\{0..N-1\}$ where N is the system size.

When an Ising system is represented by a single integer, the process of calculating the total energy of the system can be reduced to a series of bit-logic expressions. To do this expressions must be formulated to create integer values that represent the neighbours of each cell. The number of neighbours that must be calculated will be the same as the number of dimensions of the system. This is because the energy must be calculated for each bond in the system. The left bond of one atom is the same as the right bond of the other atom and thus only needs to be calculated once.

Logic expressions are also formulated to determine how many like-like bonds each atom has. In two dimensions only the cases where the atom and two like-like bonds or no like-like bonds because the only other option is to have one of each in which case the energy is 0. This process results in two integers, the first which has a 1 for every atom that has two like-like bonds and the second has a 1 for every atom with zero like-like bonds. The number of 1s in each of these integers can then be counted and multiplied by 2 and -2 respectively to give the total energy of the system. A code listing in C showing this process for a two-dimensional Ising enumeration is shown in Listing 2.

Listing 2: Bit enumeration algorithm stores the system state as an `long long` and iterates through this integer value. A series of bit masks and logic operations are used to compute the energy of the system. `__builtin_popcountl()` is a built in CPU function to count set bits in an int.

```
long long n1, n2, e1, e2, s1, s2;
for(long long i = 0; i < N; i++) {
  //Calculate neighbours
  n1 = ((i&n1_mask1) >> X) |
       ((i&n1_mask2) << ((Y-1)*X));
  n2 = ((i&n2_mask1) >> 1) |
       ((i&n2_mask2) << (X-1));
  //Calculate Energy
  s1 = (  (i^n1)  &  (i^n2));
  s2 = ((~(i^n1)) & (~(i^n2)));
  e1 = __builtin_popcountl(s1&e1_mask);
  e2 = __builtin_popcountl(s2&e2_mask);
  E = ((e1*2) - (e2*2));
}
```

In this code listing, `n1` and `n2` represent the neighbours in

the X and Y dimensions respectively. Each one can be calculated by extracting values using masks and shift operations. Then the values `s1` and `s2` are calculated which represent the number of atoms with two like-like bonds and no like-like bonds. These are calculated using XOR, AND and NOT operations. Finally built in CPU functions are used to count the number of bits in each integer and multiplies them by 2 and -2 to give the total energy. This process is shown for a 3x3 Ising system in Figure 2.

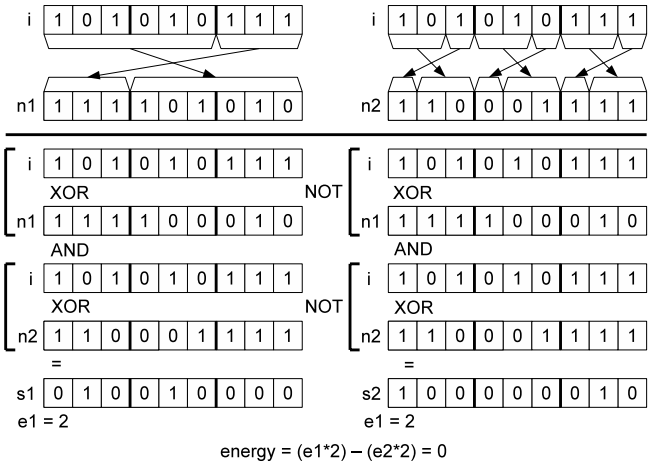

Figure 2: The process of calculating the energy of a 3x3 system state **i** in bit representation.

4.2 CUDA Implementation

The enumeration of the Ising model is a perfect candidate for parallelisation as the energy calculation for each state can be performed completely independently of the other states. The most simple approach for enumerating the Ising model on a GPU is to launch a single thread for each possible system state. Each thread's unique thread id then represents the Ising system state it is measuring in bit format. Each thread can then perform the same bit logic expressions as the CPU bit-enumeration implementation to calculate the energy of the system.

CUDA devices with compute capability 3.0 and above support a maximum of 1024 threads per block and a maximum grid size of $(2^{31}-1)$ in the X dimension and 65535 in the Y and Z dimensions [28]. This easily allows for enough threads to enumerate systems of a size that can be computed in a realistic time frame. Because the masks used in the neighbouring value calculations are exactly the same for every thread, they are stored in constant memory to minimise global memory transactions.

Once the energy of a system state has been calculated they must be collected together into the energy histogram. Due to the parallel nature of the computation this must done carefully using atomic instructions to ensure the correct values are saved in the histogram. One option that was explored was for each block to collect a local energy histogram in shared memory and then upload it to the global energy histogram once all threads in the block had finished their computation. However, in practice this was found to decrease the performance of the enumeration.

Listing 3: Bit enumeration CUDA kernel to compute the energy histogram of an Ising system. The bit masks are stored in constant memory and atomicAdd is used to update the energy histogram stored in global memory. The function `__popcll()` is a built-in GPU function to count the number of set bits in an integer..

```
__global__ void kernel(
        unsigned long long *histogram) {
  long long i = (((long long)blockIdx.x  *
                  (long long)blockDim.x) +
                  (long long)threadIdx.x)*
                   STATES_PER_THREAD;
  for(int j=0; j<STATES_PER_THREAD; j++){
    long long n1,n2,s1,s2,e1,e2,E;
    n1 = ((i&n1_mask1[0])>>1LL) |
         ((i&n1_mask2[0])<<(X-1LL));
    n2 = ((i&n2_mask1[0])>>X)   |
         ((i&n2_mask2[0])<<((Y-1LL)*X));

    //Calculate Energy
    e1 = ((i^n1) & (i^n2));
    e2 = ((~(i^n1)) & (~(i^n2)));
    s1 = __popcll(e1&c_e1_mask[0]);
    s2 = __popcll(e2&c_e2_mask[0]);
    E = ((s1*2) - (s2*2));
    atomicAdd(&histogram[E+offset],1ULL);
    i++;
  }
}
```

The final method of tuning the CUDA implementation of the enumeration is to vary the number of states each thread is responsible for computing. Rather than launching one thread for each system state, it is more efficient for each thread to compute the energies of multiple system states (especially for larger system sizes such as 6x6) as it avoids the overhead of managing and launching extra threads. For this system size it was found that computing 256 system states per thread was the most efficient and provided a performance gain of approximately 7% over computing one state per thread.

5 Performance Results

The performance results of the different enumeration implementations from Section 4 have been gathered using an 3.50GHz Intel i7-2700K using GNU gcc 4.5 with optimisations and an NVIDIA GeForce GTX680 using CUDA 5.0. The results of testing these three implementations are presented in Table 1.

From this table it can be seen that naive array implementation provides the worst performance, for the largest Ising system size tested (6x6) it took this implementation 70 min-

Q	system size	Array Enumeration time (sec)	Bit Enumeration time (sec)	CUDA Enumeration time (sec)
2	4x4	0.005129	0.002707	0.000103
	5x5	1.39604	0.571924	0.024464
	6x6	4249.92	1220.14	42.7585
	3x3x3			0.146955
	4x3x3			93.0353
	4x4x3			515215
3	3x3	0.000684		
	4x4	1.42689		
4	3x3	0.009277		
	4x4	140.629		

Table 1: Performance comparison of the 3 implementations.

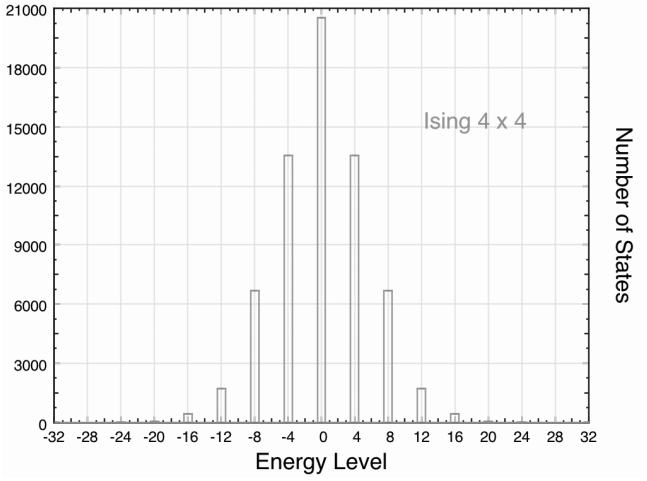

Figure 3: Energy Spectra with exact population numbers of states for 4x4 periodic Ising model

utes to complete which means it can measure approximately 16 million states per second. Compressing the system state into a single integer and using bit logic expressions reduced this time to 20 minutes which means the bit enumeration method can measure approximately 56 million states per second. Computing the bit logic expressions on a GPU using CUDA reduced the compute time to under 45 seconds and gave and average of 1.6 billion measurements per second. These results mean that enumerating the Ising model on a GPU can provide a speedup of approximately $\approx$ 100x over the array enumeration and $\approx$ 30x over the bit enumeration.

Figure 3 shows the energy histogram of all the possible states of a 4x4 Ising model. It can be seen from this figure that not all energy states are possible in an Ising system and that for a size such as this with even lengths in each dimension the histogram is symmetric. For system sizes with odd lengths the histogram will not be symmetric as it is always possible for all the states in a system to be the same but for an Ising system there is not always a configuration where every spin is different to all of its neighbours.

The number of states that fill each energy level grow very rapidly but it is also important to note the combinatorially large range of scales between the most probable energy level and that of less probable but non-empty one. For this reason it is useful to plot the populations on a logarithmic scale.

Figure 4 shows the periodic Ising models of size $N = 4^2, 5^2$ plotted on a logarithmic scale. We see the characteristic parabolic shape which agrees with other work such as that of Beale [1]. Our small system show distortions from the parabolic shape - these are due to finite size effects and manifest themselves as pinches in the shape at th side wings. It is also interesting to note that the 5^2 system is not symmetric., This is due to checkerboard pattern high energy configurations - where like like spins manage to avoid one another. Such checkerboard patterns require an even number of spins in each dimension so they are present in the spectrum for 4^2 but not 5^2.

Figure 4 also shows data for the Potts system with $Q = 3$. We see that the parabola shifts so its peak is now around energy of 10 instead of around zero. Also there is an asymmetry for both the 4^2 and 5^2 Potts systems that are linked to the odd $Q = 3$ number of states.

6 Discussion

Computational capabilities have moved on in recent years and even using a very simple brute force enumeration of states without exploiting any symmetries we can tackle Potts systems with $Q = 3, 4, ...$ that would not have been computationally feasible a decade ago. We obtained some significant performance enhancements just by packing the data structures into bit patterns instead of decomposing as conventional integers or long integers. This indicates how sensitive calculations on modern processors are to the cache and memory management system. Modern computer systems have somewhat faster clock speeds, and somewhat faster memory management systems than they did a decade ago, but neither of these two factors has improved as much as might have been hoped, extrapolating from Moore's law [26] and the previous three decades. The availability of an order of magnitude more processing cores allows a direct factor of ten in speedup as state enumeration does parallelise almost perfectly by splitting the task into independent jobs.

A key result is that we have managed to obtain nearly two orders of magnitude of performance increase using a GPU accelerator. There would appear to be significant potential in a cluster of CPUs, with multiple CPU cores and/or multiple GPUs to tackle this sort of enumeration problem.

While many researchers have invented special tricks and symmetries that can be exploited in the exact enumeration of the energy states, it is important that we not be wholly reliant on such symmetries that may not apply to other observables such as component cluster, monomers, and other lattice pattern operators.

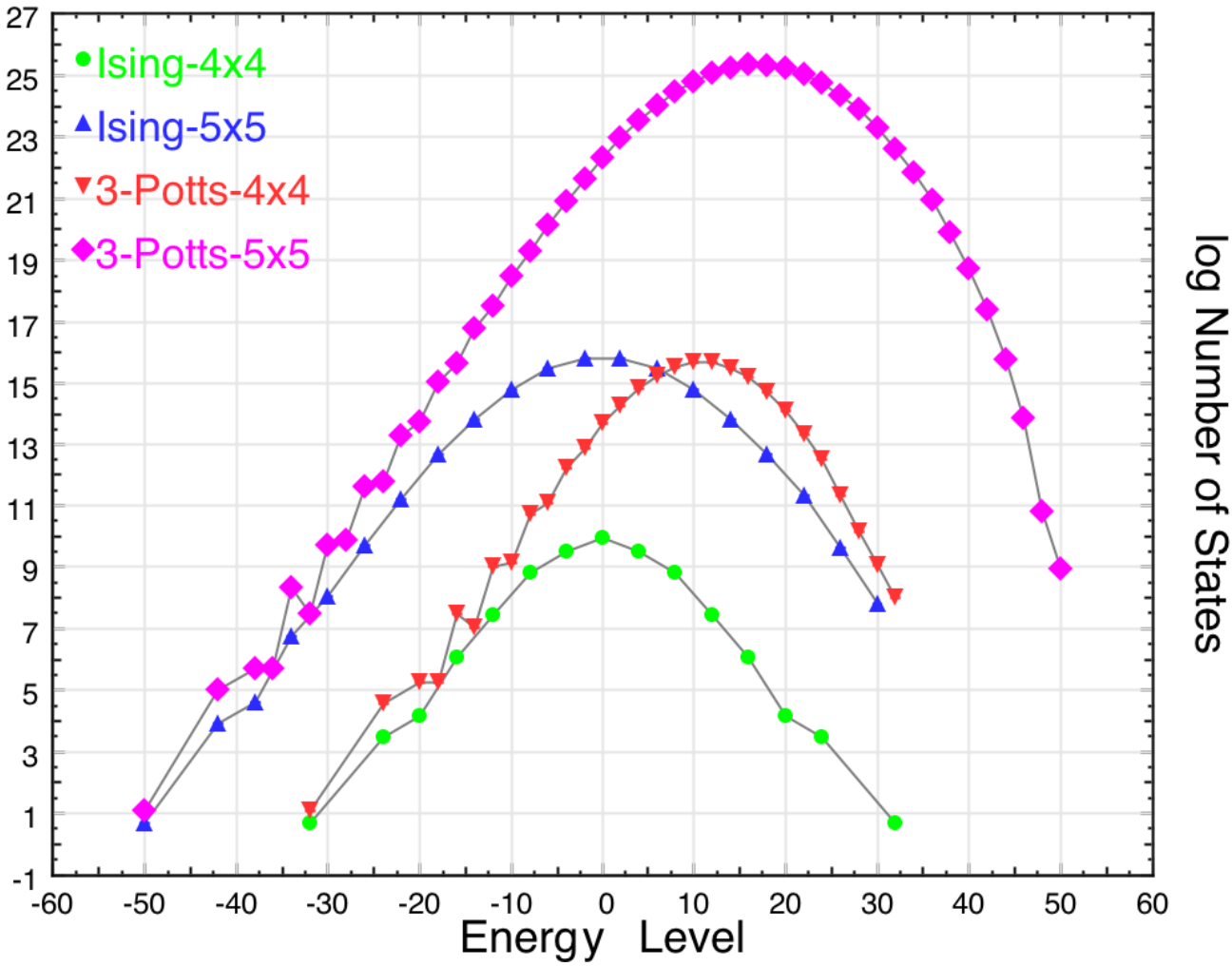

Figure 4: Energy Spectra for 2-Dimensional periodic Potts and Ising models on 4x4 and 5x5 lattices, plotted on log scale.

It is likely to be useful to identify histograms of component clusters as they are used in Monte Carlo renormalisation calculations on the Ising and related model. Components also play a significant part in cluster update algorithms such as those of Wolff [39] and of Swendsen and Wang [35]. It is likely to be instructive to determine what the relationship is between cluster size, family and scaling and so forth, and these update processes.

7 Conclusion

In summary we have explored three different methods for enumerating the Ising model on a CPU and a GPU. The bit enumeration method of provided the best performance results by reducing the amount of memory required to represent a system state and by performing the energy calculation as a series of bit-logic expressions. This enumeration method improved the performance on the CPU from 15 million states per second to 56 million states per second. The GPU provided the best overall performance and was able to measure 1.6 billion states per second, 100x faster than the CPU array enumeration method.

We have managed to obtain some preliminary results for small Potts model systems and identified interesting asymmetries and other features in the Potts exact partition function energy spectrum that are worthy of further exploration. We hope to be able to experiment with three dimensional Potts systems for $Q = 3, 4$ by parallel job decomposition of the enumeration task. We also hope to be able to incorporate fast code for identifying other oberservables such as the number of monomers and component clusters in small Ising and Potts systems.

There is also scope for future work in formulating the bit-logic expressions for computing the Potts model using the bit-enumeration method on both the CPU and the GPU as well as extending the bit-enumeration method to explore the Ising and Potts model in three-dimensions. The analysis of the model could also be extended to calculate more than just the energy of a system but also the count the number of monomers, dimers, number of clusters or their sizes.

References

[1] Beale, P.D.: Exact distribution of energies in the two-dimensional ising model. Phys. Rev. Lett. 76, 78–81 (1996)

[2] Bhanot, G., Duke, D., Salvador, R.: Finite-size scaling and the three-dimensional ising model. Phys. Rev. B 33(11), 7841–7844 (Jun 1986)

[3] Bhanot, G.: A numerical method to compute exactly the partition function with application toz(n) theories in two dimensions. J. Stat. Phys. 60, 55–75 (1990)

[4] Bhanot, G., Salvador, R., Black, S., Carter, P., Toral, R.: Accurate estimate of nu for the three-dimensional ising model from a numerical measurement of its partition function. Phys. Rev. Lett. 59, 803–806 (1987)

[5] Bhanot, G., Sastry, S.: Solving the Ising model Exactly on a 5x5x4 Lattice using the Connection Machine. J. Stat. Phys. 60(3/4), 333–346 (Nov 1990), thinking machines Preprint CS89-10

[6] Binder, K., Heermann, D.W.: Monte Carlo Simulation in Statistical Physics. Springer-Verlag (1997)

[7] Binder, K. (ed.): Monte Carlo Methods in Statistical Physics. Topics in Current Physics, Springer-Verlag, 2 edn. (1986), number 7

[8] Binder, K. (ed.): Applications of the Monte Carlo Method in Statistical Physics. Topics in Current Physics, Springer-Verlag (1987)

[9] Chang, S.C., Shrock, R.: Exact partition function for the potts model with next-nearest neighbour couplings on arbitrary-length ladders. Int. J. Mod. Phys. 15, 443–478 (2001)

[10] Chang, S.C., Shrock, R.: Some exact results on the potts model partition function in a magnetic field. Tech. rep., National Cheng Kung University, Taiwan (2009)

[11] van Dijk, W., Lobo, C., MacDonald, A., Bhaduri, R.K.: Zeros of the partition function and phase transition. arXiv 1303.4770, McMaster University, Canada (2013)

[12] Droz, M.: Noise and fluctuations in equilibrium and non-equilibrium statistical mechanics. Tech. rep., University of Geneva (2001)

[13] Fronczak, A., Fronczak, P.: Exact expression for the number of states in lattice models. Tech. rep., Warsaw University of Technology, Poland (2013)

[14] Galluccio, A., Loebl, M., Vondrak, J.: New algorithm for the ising problem: Partition function for finite lattice graphs. Phys. Rev. Lett. 84, 5924–5927 (2000)

[15] Hastings, W.K.: Monte carlo sampling methods using mrkov chains and their applications. Biometrika 57, 97–109 (1970)

[16] Hawick, K.A.: Domain Growth in Alloys. Ph.D. thesis, Edinburgh University (1991)

[17] Ising, E.: Beitrag zur Theorie des Ferromagnetismus. Zeitschrift fuer Physik 31, 253–258 (1925)

[18] Janke, W., Kenna, R.: Analysis of the Density of Partition Function Zeroes - A Measure for Phase Transition Strength, pp. 97–101. Springer (2002), computer Simulation Studies in Condensed-Matter Physics XIV

[19] Kaufman, B.: Crystal statistics ii. partition function evaluated by spinor analysis. Phys. Rev. 76, 1232–1243 (1949)

[20] Kim, S.Y.: Exact partition functions of the ising model on l x l square lattices with free boundary conditions up to l = 22. J. Korean Physical Society 62, 214–219 (2013)

[21] Kirkpatrick, S., Gelatt, C.D., Vecchi, M.P.: Optimization by simulated annealing. Science 220(4598), 671–680 (May 1983)

[22] Lee, J.H., Kim, S.Y., Lee, J.: Exact partition function zeros of a polymer on a simple-cubic lattice. Phys. Rev. E 86, 011802–1–7 (2012)

[23] Malarz, K., Magdon-Maksymowicz, M.S., Maksymowicz, A.Z., Kawecka-Magiera, B., Kulakowski, K.: New algorithm for the computation of the partition function for the ising model on a square lattice. Int. J. Mod. Phys. C 14(5), 689–694 (2003)

[24] McDonald, L.M., Moffatt, I.: On the potts model partition function in an external field. J. Stat. Phys. 146, 1288–1302 (2012)

[25] Metropolis, N., Rosenbluth, A.W., Rosenbluth, M.N., Teller, A.H., Teller, E.: Equation of state calculations by fast computing machines. J. Chem. Phys. 21(6), 1087–1092 (Jun 1953)

[26] Moore, G.E.: Cramming more components onto integrated circuits. Electronics Magazine April, 4 (1965)

[27] Nguyen, H.C., Berg, J.: Mean-field theory for the inverse ising problem at low temperatures. Phys. Rev. Lett. 109, 050602–1–5 (2012)

[28] NVIDIA: CUDA C Programming Guide, 5.0 edn. (October 2012)

[29] Pearson, R.B.: Partition function of the ising model on the periodic 4x4x4 lattice. Phys. Rev. B 26, 6285–6290 (1982)

[30] Potts, R.B.: Some generalised order-disorder transformations. Proc. Roy. Soc pp. 106–109 (1951), received July

[31] Raychaudhuri, S.: Kinetic Monte Carlo Simulation in Biophysics and Systems Biology (2013)

[32] Sangaranarayanan, M.V.: Two-dimensional ising model in a finite magnetic field for the square lattice of infinite sites-partition function and magnetization. arXiv 1302.1084, Indian Institute of Technology, Madras (Jan 2013)

[33] Stanley, H.E.: Introduction to phase transitions and critical phenomena. Oxford Science Publications (1987)

[34] Strasberg, P., Schaller, G., Brandes, T., Esposito, M.: Thermodynamics of a physical model implementing a maxwell daemon. Phys. Rev. Lett. 110, 040601–1–5 (January 2013)

[35] Swendsen, R.H., Wang, J.S.: Nonuniversal critical dynamics in Monte-Carlo simulations. Phys. Rev. Lett. 58(2), 86–88 (Jan 1987)

[36] Vernier, E., Jacobsen, J.L.: Corner free energies and boundary effects for ising, potts and fully-packed loop models on the square and triangular lattices. arXiv 1110.2158, Ecole Normale Superieure, France (2011)

[37] Vinothkumar, M., Nandhini, G., Sangaranarayanan, M.V.: Partition function of the two-dimensional nearest neighbour ising models for finite lattices in a non-zero magnetic field. J. Chem. Sci. 124, 105–113 (2012)

[38] Wohlfarth, E.P.: Theory and experiment in metallic magnetism. J. Magn. and. Magn. Mat. 45, 1–8 (1984)

[39] Wolff, U.: Collective Monte Carlo Updating for Spin Systems. Phys. Lett. 228, 379 (1989)