A new algorithm for the Ising problem

Anna Galluccio
Instituto di Analisi dei Sistemi ed Informatica - CNR
viale Manzoni 30, 00185 Roma, Italy
galluccio@iasi.rm.cnr.it

Martin Loebl
Department of Applied Mathematics
Charles University
Malostranské nám. 25, 11800 Praha, Czech Republic
loebl@kam.ms.mff.cuni.cz

Jan Vondrák
Department of Applied Mathematics
Charles University
Malostranské nám. 25, 11800 Praha, Czech Republic
vondrak@kam.ms.mff.cuni.cz

January 27, 2000

Abstract

We present a new efficient method to find the Ising problem partition function for finite lattice graphs embeddable on an arbitrary orientable surface, with integral coupling constants bounded in the absolute value by a polynomial of the size of the lattice graph. The algorithm has been implemented for toroidal lattices using modular arithmetics and the generalized nested dissection method. The implementation has substantially better performance than any other as far as we know.

1 Max Cut and the Ising model

There are many open questions about the effects of disorder and frustration in spin-glasses. The simplest spin-glass system used to study these questions numerically is the Edwards-Anderson model. Its Ising version may be described as follows: A graph (or lattice) is a pair $G = (V, E)$ where V is a finite set of *vertices* and E is a set of unordered pairs $\{u, v\} \subset V$ (*edges*). A *coupling constant* J_{ij} is assigned to each edge $\{i, j\}$; this factor characterizes the interaction between particles represented by i and j. A physical state of the system is an assignment of spin $\sigma_i \in \{+1, -1\}$ to each vertex i. The *Hamiltonian* (or energy function) is then defined as $H(\sigma) = -\sum_{\{i,j\}\in E} J_{ij}\sigma_i\sigma_j$. The distribution of physical states over all possible energy levels is encapsulated in the *partition function* $Z(\beta) = \sum_\sigma e^{-\beta H(\sigma)}$ from which all fundamental physical quantities may be derived.

The partition function is very close to the *generating function of cuts* which is a standard concept in graph theory. For a subset of edges $\alpha \subset E$, $w(\alpha)$ means the sum of the weights $w(e)$ associated with the edges in α.

A *cut* of a graph $G = (V, E)$ is a partition of its vertices into two disjoint subsets $V_1, V_2 \subset V$, and the implied set of edges between the two parts: $C(V_1, V_2) = \{\{u, v\} \in E : u \in V_1, v \in V_2\}$

The *generating function of cuts* is a polynomial $\mathcal{C}(G, x)$ which equals the sum of $x^{w(C)}$ over all cuts C of G.

Min Cut and *Max Cut* are combinatorial optimization problems which correspond to determining ground state energy and maximum possible energy of the Ising model:

Given a graph, divide its vertices into two parts so that the number of edges between them is as small as possible (as large as possible). More generally, the edges may have arbitrary weights, and we need to minimize or maximize the sum of weights over all the edges between the two sets of vertices.

These combinatorial problems have their own history, but what makes them so widely studied is the enormous number of applications they find in different fields. The general Max Cut problem has defied efficient solution so far, and indeed, it was proved to be *NP-hard* [GJ], even in the case when all edge weights are equal to 1 or -1. In spite of that, many attempts have been made to tackle the problem with approximation and randomized algorithms ([DP], [PR], [GW]).

Polynomial time methods for toroidal square lattices were suggested in the early 60's by Kasteleyn [Kast, Kast1], and Kac and Ward [KW]. Kac and Ward tried to calculate the partition function as a determinant of a $4n \times 4n$ matrix over complex numbers and even though their original derivation was not quite exact, it showed that such an approach was indeed possible. A similar method was used by Kardar and Saul [Saul] to obtain the partition function of a toroidal spin glass in estimated time $O(n^{3+\epsilon})$, $\epsilon < 1$. They asked whether an efficient method exists for general toroidal graphs. (This paper answers the question affirmatively.) Another polynomial time algorithm to solve the Max Cut problem in toroidal square lattices was proposed by Barahona in [B1] and in an unpublished manuscript [B2].

There are branch-and-cut algorithms which can be applied effectively to the case of toroidal square lattices. Barahona et al. [BGJR] and De Simone et al. [DJRR1], [DJRR2] used integer programming to solve large instances of the Max Cut problem for toroidal square lattices, with general weights or with weights $+1$ and -1 only. The method of De Simone et al. [DJRR2], which has been so far the most successful one to find exact spin glass ground state, works in estimated time $O(n^3)$ for toroidal square lattices of n vertices, for $n \leq 50$. The method is, of course, non-polynomial for general n.

However, there has been no deterministic algorithm which would produce an exact result in polynomial time for any toroidal graph, for example. Due to a recent result proved by two of the authors [GL6], though, it has become possible to solve Max Cut in polynomial time for any graph of genus bounded by a constant. The method produces actually the generating function of cuts which contains information about all possible cuts in the graph. In terms of statistical physics, this means the distribution of physical states over all possible energy levels of the system. This can be of tremendous interest for anyone studying a particular Ising model.

The aim of this paper is to describe the features of the algorithm and its implementation with emphasis on the toroidal spin-glass problem. The algorithm is parallel in nature and it is based on modular arithmetic calculations. It may be particularly useful for calculations of energy and degeneracy of the ground-state and first excited state. It makes it possible, by using massive parallelization, to determine these quantities exactly for lattices of size up to (100×100).

2 Theory behind the algorithm

An *eulerian subgraph* of graph $G = (V, E)$ is a set of edges $U \subset E$ such that each vertex is incident with an even number of edges from U. The *generating function of eulerian subgraphs* is a polynomial $\mathcal{E}(G, x)$ which equals the sum of $x^{w(U)}$ over all eulerian subgraphs U of G. There is a duality between the partition function and the generating function of eulerian subgraphs, which was discovered by van der Waerden [vdW]:

$$Z(\beta) = 2^n \prod_{\{i,j\} \in E} \cosh(\beta J_{ij}) \quad \mathcal{E}(G', \tanh(\beta J_{ij}))$$

In other words, the partition function can be expressed as the generating function of eulerian subgraphs of the same graph, with modified edge weights $w'_{ij} = \tanh(\beta J_{ij})$.

A *perfect matching* of graph $G = (V, E)$ is a set of edges $P \subset E$ such that each vertex is incident with exactly one edge from P. The *generating function of perfect matchings* is a polynomial $\mathcal{P}(G, x)$ which equals the sum of $x^{w(P)}$ over all perfect matchings P of G. The generating function of eulerian subgraphs of a graph G may be transformed into the generating function of perfect matchings of a modified graph G_s. We use Fisher's construction described in [Fish2]. It is local in the sense that it only modifies each vertex in a way dependent on its degree and it preserves the genus of the graph.

Hence we need to describe an efficient way to compute the generating function of perfect matchings of a graph with bounded genus.

Let $G = (V, E)$ be a graph with $2n$ vertices, $\{x_e; e \in E\}$ the variables assigned to edges and D an orientation (a fixed ordering of the two vertices of each edge). Let $A(D)$ denote the antisymmetric adjacency matrix where $a_{ij} = x_{(i,j)}$, if (i, j) is a directed edge, $a_{ij} = -x_{(j,i)}$, if (j, i) is a directed edge and $a_{ij} = 0$, if $\{i, j\}$ is no edge.

The Pfaffian of this matrix is defined as

$$Pf(A(D)) = \sum_P sgn(P) a_{i_1 j_1} a_{i_2 j_2} \ldots a_{i_n j_n}$$

where the sum is taken over all partitions of the index set $\{1, 2, \ldots, 2n\}$ into pairs $i_1 < j_1$, $i_2 < j_2$,...,$i_n < j_n$ and $sgn(P)$ is the sign of the permutation $(i_1, j_1, i_2, j_2, \ldots, i_n, j_n)$.

It can be observed that the non-zero terms contributing to the Pfaffian are exactly those corresponding to perfect matchings of G (partitionings of the set of vertices into pairs where there is an edge between each pair). However, each of them comes with a positive or negative sign, depending on the orientation D. What we would like to find is a special orientation which produces positive signs for all perfect matchings. Then the Pfaffian would be exactly equal to the generating function of perfect matchings. Such a special orientation (called *Pfaffian orientation*) exists indeed for planar graphs, which was proved by Kasteleyn [Kast].

For general graphs we cannot rely on the Pfaffian orientation, as it may not always exist. However, the situation is solved by the *Galluccio-Loebl theorem* [GL6] which states that a graph G of genus g has 4^g *relevant orientations* such that a suitable linear combination of the corresponding Pfaffians yields the generating function of perfect matchings of G. To calculate the Pfaffians we use a method which was developed by Alan George [Geo] and later refined by Lipton, Rose and Tarjan [LRT]. Their approach, though intended for Gaussian elimation, can be geared to Pfaffian elimination. For graphs G of bounded genus, Pfaffian elimination on matrix $A(D)$ where D is an orientation of G and G has n vertices can be performed in $O(n \log n)$ space and $O(n^{\frac{3}{2}})$ time.

Here it should be remarked that the units of the time complexity are actually the basic operations with elements of the matrices. However, these elements are polynomials in one variable. If we performed the operations symbolically, we could hardly hope to carry them out in constant time. Also, the memory requirements for symbolic manipulation would be enormous. Instead, we decided to perform the elimination numerically, and furthermore, in some finite field. This ensures that the operations can be performed in constant time and the elements take up constant space.

On the other hand, the result of this computation is only the value of the generating function modulo some prime. In order to reconstruct the whole polynomial, we need to calculate the values at sufficiently many points, obtain the polynomial by interpolation in the finite field, and then (if necessary) repeat the computation in several finite fields and compose the partial results to get the complete polynomial.

Another advantage of this approach is that the algorithm is parallel in nature - the computation is broken up into many independent parts which can be processed in parallel. Or we can actually use the coefficients obtained in one finite field as partial results; this can be useful if we only want to find the value of the ground state energy, i.e. the first non-zero coefficient in the polynomial.

As a consequence, the partition function can be computed efficiently and precisely for any graph of bounded genus.

3 Algorithm overview

Let us summarize the algorithm to calculate the partition function for a given lattice graph $G = (V(G), E(G))$.

1. *Find prime numbers* $p_1, p_2, ..., p_k < 2^{16}$ *so that* $\prod_{i=1}^{k} p_i > 2^{|V(G)|}$.

For each of them, repeat the remaining steps of the algorithm, performing all operations in $GF[p_i]$.

2. *Choose $m+1$ distinct elements $x_0, ..., x_m \in GF[p_i]$, where m is the maximum possible degree of the generating function. (Avoid $x_j = 0$ because the elements must be invertible). For each of them, repeat the following step.*

3. *Construct the matrices encoding the relevant orientations of the modified graph G_s, with hyperbolic functions $\tanh(x, -\frac{w_{ij}}{2})$ substituted for edge weights. For every occurrence of $x^{-\frac{1}{2}}$ substitute $x^{-\frac{1}{2}} = x_i$ and calculate the four Pfaffians. From these values, calculate the value of $\mathcal{C}(G, x_i^{-2})$.*

4. *Obtain the coefficients of $C(G, x)$ (modulo p_i) by interpolation in $GF[p_i]$; in other words, if we can supply the values of our polynomial modulo some prime p, we can also extract its coefficients modulo p.*

5. *Apply the Chinese remainder theorem and compose the results for each coefficient from all the finite fields, to obtain the complete partition function.*

Notes on the complexity of the algorithm:

- The number of finite fields is $O(n)$, assuming there is a sufficient number of primes in the range where our hardware is able to perform modular arithmetics in constant time. Due to this constraint, the algorithm is actually unable to work properly for an arbitrarily large input, and any complexity analysis in the sense of asymptotic behavior is meaningless. On the other hand, the limit on the size of lattices we are able to process is well beyond our practical possibilities. The product of all primes below 2^{16} is approximately $2^{2^{16}}$, which means we can process lattices with at most 2^{16} vertices. Such a computation would last $10,000$ years on a typical PC. If this should prove insufficient, we could still move to 32-bit arithmetics and the limit of 2^{32} vertices which would be enough to keep all our computing resources busy for more than the lifetime of our universe.

- If the edge weights are bounded by a constant, the number of evaluations in each of the fields is $O(n)$. A single evaluation of the polynomial takes $O(n^{\frac{3}{2}})$, so the computation in each finite field takes $O(n^{\frac{5}{2}})$ time. The total time complexity of Step `3.` (under the restrictions mentioned above) is therefore $O(n^{\frac{7}{2}})$.

- The interpolation in Step `4.` and the final composition in Step `5.` take $O(n^3)$ time in total, so they are faster than the Pfaffian evaluation in Step `3.`

- Steps `2.`, `3.` and `4.` can be parallelized easily. The computation in each of the finite fields can be performed separately, and the communication is trivial - in Step `5.`, we only have to send the results modulo each of the primes; i.e., data of size $O(n^2)$. With this degree of parallelization ($O(n)$ processors are needed), Steps `2.`, `3.` and `4.` take $O(n^{\frac{5}{2}})$ time, whereas Step `5.` takes $O(n^3)$. To remove this obstacle, we could parallelize it as well; every processor would produce one of the $O(n)$ coefficients in $O(n^2)$ time. Then the total (parallel) time complexity would be $O(n^{\frac{5}{2}})$.

- As mentioned above, the computation in each of the finite fields takes $O(n^{\frac{5}{2}})$ time. The product of this part of the algorithm is the partition function with coefficients modulo p_i. The information we obtain here is significantly reduced, but we can still use it for detection of the ground state energy (which correspond to the first non-zero coefficient in the partition function):

  If a coefficient modulo a randomly chosen number is non-zero, then the coefficient is certainly non-zero. On the other hand, if the remainder is zero, the original coefficient is zero with high probability. By allowing ourselves to choose from a wide enough range of primes, we can make the chance of error arbitrarily small. We can also reduce the probability of error by performing independent computations in several finite fields.

  The time complexity $O(n^{\frac{5}{2}})$ in one finite field compares favourably with the estimated complexity $O(n^3)$ (for toroidal grids of sizes up to (50×50) only) of the most successful method using integer programming technics of [DJRR2], mentioned in the introduction.

- The complexity of our implementation is comparable with the estimated complexity of the implementation of Kardar and Saul [Saul]. Apart from our method, this is the only method for the exact calculation of the generating function of cuts (for toroidal square lattices) we know about. However, the advantages of our algorithm are its self-contained structure and easy parallelization. Moreover, as mentioned above, computations in one or a few finite fields yield the values of some coefficients with high probability.

The graphs which seem suitable as models for many physical phenomena in crystallic structures are *toroidal lattices* [Fish]. For toroidal lattices we can use further optimization. In order to test the software we carried out a number of experiments on random lattices. For illustration, the running times (sequentially on a Pentium/200 MHz machine) are as follows:

The complete partition function for a 10×10 grid takes 40 seconds to compute. A 20×20 lattice takes 1 hour, and a 30×30 lattice 20 hours, but the computation in one finite field (which is practically sufficient for ground state detection) takes only 3 or 20 minutes, respectively. A 50×50 lattice takes 6 hours in one finite fields and 40 days in total.

Acknowledgement.

We would like to thank Jirka Matoušek for helpful discussions and in particular for drawing our attention to the modular arithmetics method. We would also like to thank Giovanni Rinaldi for enlightening discussions and continuous support.

References

[B1] F. Barahona. On the computational complexity of Ising spin glass models. *J. Phys. A*, 15, 1982.

[B2] F. Barahona. Balancing signed toroidal graphs in polynomial time. *Preprint University of Chile*, 1983, Unpublished manuscript.

[BGJR] F. Barahona, M. Grötschel, M. Jünger, and G. Reinelt. An application of combinatorial optimization to statistical physics and circuit layout design. *Operations Research*, 36:493–513, 1988.

[Fish] Sora Cho and Matthew P. A. Fisher. Criticality in the 2D random-bond Ising model. *Physical Review B*, 77, 1997.

[DJRR1] C. De Simone, M. Diehl, M. Jünger, P. Mützel, G. Reinelt, and G. Rinaldi. Exact ground states of Ising spin glasses: New experimental results with a branch and cut algorithm. *Jour. of Stat. Phisycs*, 80:487–496, 1995.

[DJRR2] C. De Simone, M. Diehl, M. Jünger, P. Mützel, G. Reinelt, and G. Rinaldi. Exact ground states of $2d \pm j$ Ising spin glasses. *Jour. of Stat. Phisycs*, 84:1363–1371, 1996.

[DP] C. Delorme and S. Poljak. Laplacian eigenvalues and the maximum cut problem. *Math. Prog.*, 62:557–574, 1993.

[Fish2] M.E. Fisher. On the dimer solution of planar Ising models. *Journal of Mathematical Physics 7*, 10, 1966.

[GL6] A. Galluccio and M. Loebl. A theory of pfaffian orientations I: Perfect matchings and permanents. *Electronic Jour. Combinatorics*, 6(1), 1999.

[GJ] M. R. Garey and D. S. Johnson. *Computers and Intractability.* W. H. Freeman and Company, New York, 1979.

[Geo] A. George. Nested dissection of a regular finite element mesh. *SIAM J. Numer. Anal.*, 10:345–363, 1973.

[GW] M. X. Goemans and D. P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *J. of ACM*, 42:1115–1145, 1995.

[alp] M. Junger, G. Reinelt, H. Rieger, and G. Rinaldi. *Algorithmic Techniques in Physics.* Dagstuhl-Seminar-Report 197, Dagstuhl, 1997.

[KW] M. Kac and J.C. Ward. A combinatorial solution of the two-dimensional Ising model. *Physical Review*, 88, 1952.

[Saul] M. Kardar and L. Saul. The 2D $\pm$ Ising spin glass: exact partition functions in polynomial time. *Nuclear Physics B*, 432:641–667, 1994.

[Kast1] P. W. Kasteleyn. Dimer statistics and phase transitions. *Jour. Math. Physics*, 4:287–293, 1963.

[Kast] P.W. Kasteleyn. Graph theory and crystal physics. *Graph theory and theoretical physics*, New York, 1967. Academic Press.

[LRT] R.J. Lipton, D.J. Rose, and R.E. Tarjan. Generalized nested dissection. *SIAM J. Numer. Anal.*, 16:346–358, 1979.

[PR] S. Poljak and F. Rendl. Solving the max-cut problem using eigenvalues. *Discr. Appl. Math.*, 62:249–278, 1995.

[vdW] B.L. van der Waerden. Die lange reichweite der regelmassigen atomanordnung in mischkristallen. *Z. Physik*, 118:473, 1941.