Superlinear Speedup of Parallel Calculation of Finite Number Ising Spins Partition Function

Nefedev K.V.[1,2], Peretyatko A.A[1]

[1] *Far Eastern Federal University,School of Natural Science, Sukhanova str. 8-43, Vladivostok, Russia*
[2]*Far Eastern Branch Russian Academy of Science, Institute of Applied Mathematics, Radio str. 7, Vladivostok, Russia*

nefedev.kv@dvfu.ru, peretyatko@phys.dvgu.ru

Abstract. *The high-performance parallel algorithm for rigorous calculation of partition function of lattice system with finite number Ising spins was developed. The parallel calculations run by C++ code with using of Message Passing Interface. The superlinear speedup was obtained for given executing code. The reasons of experimentally observed Amdahl's law violation are considered.*

Keywords

Parallel algorithmization, Ising model, partition function, superlinear speedup, Amdahl's law, high performance computing.

1 Introduction

Among the huge variety of fundamental scientific tasks there are problems from the solution of which ones the comprehension progress of the Nature depends in whole. The solution for Ising model, a simple mathematical model for phase transitions in correlated systems, could be such a problem. The rigorous solution in this model it is necessary not only for the development of a statistical physics, but also and in related areas: nanotechnology, nanomaterial science, nanobiotechnology and et al. Today it is possible to work within systems which ones consist of relative small (finite) number of atoms. The exact solution in Ising model would be very desirable for further development of science.

The existing modern single-threaded searching methods have strong limitations and not allow obtain the solution within a reasonable period of time. The total amount of a terms partition function grows up as power function with the increasing of the number of elements in the system, moreover Ising model can't be solved exactly analitically for 3D case. This task it is possible refer to NP-complete problem [1-2]. Thus for exact construction of the partition function even for small finite systems it is extremely desirable to have the scalable algorithm and to use the computer power, that could ensure the intensive parallel calculation. The productivity of computers enhances every ten years approximately in thousand times, that essentially expands the class of solvable tasks and possibillities of algorithms.

Approximate research numerical methods of 2D and 3D Ising model, such for example as Metropolis algorithm and method Monte Carlo (MC) [3-4] have get wide popularity. With respect to rigorous solution the situation is more problematical. The rigorous analytical solution for 1D case was obtained by Ising in 1925, and exact solution for 2D case was obtained by Onsager in 1944 [5] (difference between "exact" and "unrigorous" defined in [6]). In work [7] by authors it was shown that the critical temperatures of transition of finite systems to ferromagnetism could be calculated more precisely than in approximate analytical theories. Therefore it is interesting to construct the rigorous solution for finite number of spins for a lattices with the number of nearest neighbors Z or more, as well as the comparison of results, which ones can be obtained by means of numerical simulation and method rigorous calculation.

We have developed the high-performance scaled parallel algorithm of rigorous calculation of partition function for the finite number N of Ising spins with ferromagnetic exchange on the planar lattice with Z=4, and show the possibility of generalization of this algorithm on 3D arbitrary lattices. The measurement of time execution of parallel C++ code shows superlinear dependency of speedup.

Two main reasons of superlinear speedup are connected with two possible ways of parallelism: with parallelism over data and parallelism over commands. First one is the caching effect. If in result of code parallelization the working set of data is reduced, that all data can be placed in cache memory and, as consequence, a time of accesses to memory will be reduced. Other reason of superlinear speedup is the decreasing of common number of steps in parallel program in comparison with sequential analog. In addition, it is not possible except a simultaneous realization of these

preconditions for Amdahl's law violation. In this paper we present results of measurements of execution time and consider possible reasons for observed phenomenon.

2 Ising Model

Let us shortly remind the main idea of the simplest model of ferromagnetism - the Ising model, which one has not rigorous solution yet in spite of all its simplicity. In this model it permits only the interactions between nearest neighbors usually. The Hamiltonian of the system in general case is

$$H = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{Z} J_{ij} S_i S_j - h\sum_{i=1}^{N} S_i \qquad (1)$$

where h is external magnetic field, the summation over j denotes the summation over neighbors (nearest here). Using summands in the (1) it is possible to put for the each configuration of 2^N possible configurations in correspondence two numbers – the energy and the spin excess [8]. Exchange constant J_{ij} is positive and in considered case equals to one (in principle could be sign-changing in general case). The partition function of finite number of Ising spins

$$Z_N(h,T) = \sum_{S_1}\sum_{S_2}\ldots\sum_{S_N} Exp[-H/T] \qquad (2)$$

The troubles with the rigorous numerical calculation of (2) are in the estimation of the degeneracy multiplicity of the spin excess M_i (total sum of all S_i) over energy E_i (total sum exchange energy of all pair spins). This value could be estimated as the number of combinations with given quantity of pairs of side by side "units" and "zeros" and taking into account the periodical boundary conditions. In other words in the combinatorial terms we need estimate the number of permutations with repeating and limitations on the distribution and size of equal element groups. Total amount of degeneracy multiplicities fixed M_i over all permitted E_i is the binomial coefficient.

3 Scalable parallel algorithm

We have elaborated high performance scalable parallel algorithm of calculation of the partition function Ising 2D lattice. The simplicity of proposed approach gives the possibilities of extension on 3D lattice with given number of a nearest neighbors *Z*. The limitation of linear search algorithm is in linear dependence from computer performance.

In general case the lattice can be presented by the set of bit-vectors. For 2D this set bit-vectors $\{a_i\}$ is one dimension array, where two nearest elements of it correspond in two nearest columns of lattice, for 3D it is two dimensional matrix $\{a_{ij}\}$. De facto the information about distribution of spin states each column of the lattice was written it the integer variables, number of which ones equals to linear dimension of lattice. Integer variable size is four bytes, i.e. 32 bits. A unary and pair logical as well as arithmetical operations with a parts of such bit-vectors, the number of significant bits which ones equal to second linear dimension of lattice, give us of M_k and E_k. Total sum of M_k and E_k of all variables allow obtains the M_i and E_i for one configuration. All configurations can be "counted" by sequential changing of value of each integer variable in a diapason from 0 to 2 in power *n*, where *n* is a linear dimension of the lattice.

3.1 Spin excess

The sum of all units of all bit-vectors gives us the number of spins, which ones have direction "up". It is easy to define the spin excess as the deference between the total number of spins and double the amount of the spin "up". The quantity of units was defined with using of Kernighan algorithm [9]

```
int q_units(unsigned short int i)
{ int j = 0;
  while (i)
  {i&=i-1; ++j;}
  return j;
}
```

Really, in code of program we have used the possibility of optimization which one connects with storage in an unsigned short integer cache array $\{b_i\}$ of units number (weights of bit-word) calculated once for each decimal numbers and reference to the elements of the array in cycles. The reference to the elements of the given array is used instead of an all arithmetical operations, where the enumeration of the bit-vector units takes place.

3.2 Rows exchange energy

The calculation of exchange energy between spins in rows can be done in three steps:

1) The operation XOR between all pairs a_i and a_{i+1}, including periodical boundary conditions $a_1 \wedge a_N$, for 2D lattice ($a_{ij} \wedge a_{i+1,j}$ or $a_{ij} \wedge a_{i,j+1}$ for 3D case);
2) The invert conversion of the previous operation result, i.e. implementation of NOT operation;
3) The nullification of bits which ones have not information about spin state of the system. The subtraction from resultant bit-vector 2^{32}-2^n (order "32" is for "int" type of the bit-word size);
The quantity of units in the resultant bit-word gives us the number of the pair exchange interactions with positive sign in rows, and should be stored in the separate variable E_n.

3.3 Columns exchange energy

The enumeration of the positive pair exchange spins interactions in the each a_i (a_{ij}) column achieve by the following way:
1) To each decimal number "j" - index of the integer aray $\{c_j\}$ - put in correspondence the dicimal number c_j, the binary notation of which one is circular shift of significant bits of number "j";
2) The implemention of the operation XOR between all pairs of integer number a_i and c_j,where $j= a_i$;
3) The invert conversion (NOT operation) of the previous operation result;
4) The nullification of bits which ones have not information about spin state of the system;
5) The references on elements of the array $\{b_i\}$ is used for the determination of units quantity and summation with En;
The final value of *En* is total amount of the all pair exchange interactions with positive sign for one configuration. There are not troubles to calculate the total exchange energy of the spin pairs interactions by using of the simple arithmetical expression $E_{tot}=2(-En+N)$.

3.4 Parallelism and superlinear speedup

The Massage Passing Interface (MPI) standard extends the possibility of computers and makes available parallel algorithmization that allows solving the tasks, the solution of which ones couldn't be obtained previously in the reasonable time. The possibility of the parallel execution in our C++ code is implemented via the MPI libraries.

The number the nested loops equals to the length of the vector $\{a_i\}$, i.e. number bit-vectors. The task of the parallelization of the nested iterations requires the application of the technology of dynamical generation processes. In the current version of the program we have not used the dynamical parallelization, but there are some possibilities of the organization of a dynamical generation of the parallel processes.

The elaborated algorithm for the rigorous calculation of partition function for finite number of Ising spins on the square lattice displays the excellent scalability. Each process runs approximately the same time. There is the superlinear dependence of speedup from number of cores. This phenomena is more denominate for greater number of spins. The data for different number of spins (2^{30} and 2^{36} configurations) is very close to each other. The small dispersion for greater number of cores connects with the short times of calculation and the ratio increasing of the message passing time to time of calculation, Fig.1.

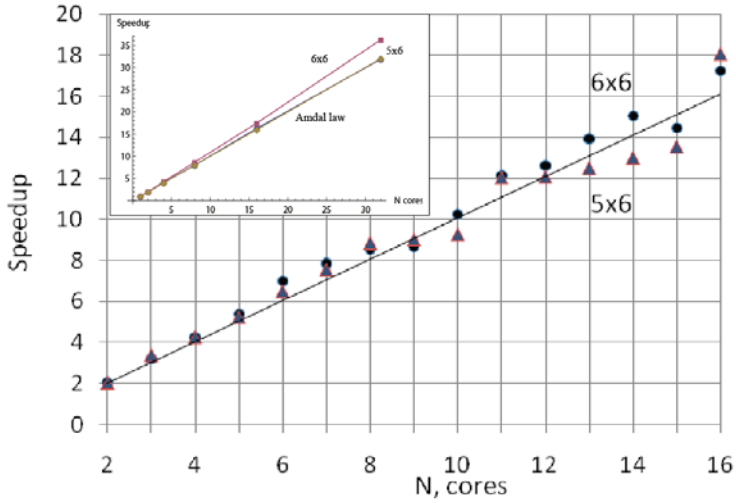

Fig. 1. The speedup for 36 and 30 spins on the lattice in dependence on the number of cores. The circles – 6x6 and the triangles – 5x6 spins, solid line is Amdal's law. On the insertion there are the execution times of other numerical experiments also and for the 32 cores of the HP cluster.

The total accuracy of the computation was controlled by using of the total known number configurations, of the binomial coefficients, also by other well-known parameters, which ones can be calculated directly for small number of spins.

Tab 1. The measuring of process durations and speeding up by means of different methods for 5x5 lattice system. Time in sec.

		clock();	MPI_ Wtime();	ftime();	System Command "time"	open-closing datafile
The running time of the process #	**0**	37.910	37.9297	38.662	38.613	37.478
	1	37.460	37.4688	37.468		
	2	37.710	37.7188	37.66		
	3	37.530	37.918	38.612		
Single process		156.280	156.312	156.315	156.536	156.286
Speedup 4 nodes		4.122	4.121	4.043	4.054	4.170
Speedup 8 nodes		8.068	8.064	8.062	7.714	8.08
Speedup 16 nodes		8.247	8.279	8.279	7.529	8.116

For given algorithm the measurment of the speedup showed the superlinear values for small number of processes. The running time of application was estimated with using of the library *time.h* and the MPI inline functions *clock(), MPI_Wtime(), ftime(),* system command "*time*" and duration writing of *datafile,* see Tab. 1. It is observed the growthing of superlinear speedup with increasing of cores amount.

The parallelism provides decomposition of data blocks, which ones are send to CPU cores for calculations. This promotes that size of block was less then volume of cache memory core, therefore it was fully placed in cache. During work of sequential code a data had to always passed through cache memory, and processor was compeled to wait of necessary data appearance in cache. For sequential code a volume of the data was relatively large for displacement of previous cache lines, therefore any using of given lines of cache demand a waiting time of data. In parallel verstion all data were divided on blocks which ones are placed in cach of core as whole, thereby after loading of all cache lines a time costs on a waiting were not need. The using of some cores eliminates cost, which were connected with a sequential execution on the one core.

Other words requirements of our initial task to calculation resources exceeded the possibility of the one processor. The execution of the parallel code on some processors has brought, that subtasks with a less volume were placed in memory, therefore requirements to resources (cache, memory and etc.) were reduced. This was the main reason of superlinear speedup. The considerable reducing of a cach transaction operation quantity led to removal of time costs on an execution of the parallel program code in relative to sequential verstion.

Amdahl's law supposes, that a number of operations, which ones need for a receiving of result, and also volume of necessary operation for calculations remain the same. The existence of superlinear speedup can be easy explained if this assumption does not satisfy. If a volume of instructions and data of sequential and parallel codes are different, that Amdahl's law cannot be used, since it sets the limit and restrictions on productivity of parallel algorithms of the same classes without taking into account the architecture of multiprocessor and features of caching algorithms.

4 Conclusion

The high-performance parallel algorithm for rigorous calculation of partition function of the lattice systems of finite number Ising spins was developed. The algorithm is scalable and it is possible to extend it for the running on huge calculation recourses. Superlinear growth of the speedup in dependence on computer powers is obtained. The nested iterations can be executed with utilization of the approach dynamical generation of the processes.

Even with linear increasing of speedup by existing methods it is impossible to calculate M_i and E_i for each configuration of 10x10 system i.e. 100 spins. So far as 2^{100} is approximately 10^{33}, therefore for computational cloud with computer speed in order ExaMips (10^{18} Mips) only for the summation until decillion it is necessary 10^{15} seconds or ~31,7 millions year! The same time needed for the 3D Ising lattice with structure 4x5x5. Of course, the availability of generating function could simplify the solution, but today we have not rigorous solution even for 2D Ising lattice. The future approach to optimization of the chosen scheme of calculation of partition function could be done by means of taking into account the high symmetry of the this task. Reasons of superscaling or superlinear speedup could be in organization of low-level programming algorithms and interactions of processor with memory devices. This is very interesting phenomena, which one demands additional researches.

5 Acknowledgments

This work supported by Scientific Fund of Far Eastern Federal University, Project #12-07-13000-18/13.

References

[1] Istrail, Sorin (2000), "Statistical mechanics, three-dimensionality and NP-completeness. I. Universality of intractability for the partition function of the Ising model across non-planar surfaces (extended abstract)", Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing, ACM, pp. 87–96, MR2114521
[2] Barry A. Cipra, "The Ising model is NP-complete", SIAM News, 2000, Vol. 33, No. 6.
[3] N. Metropolis, S. Ulam The Monte Carlo Method, — J. Amer. statistical assoc. 1949 44 № 247, pp. 335—341.
[4] Neumann J., "NBS Appl. Math, scries", 1951 № 12, p. 36-38.
[5] Martin Niss, History of the Lenz-Ising Model 1920–1950: From Ferromagnetic to Cooperative Phenomena Archive for History of Exact Sciences, Volume 59, Number 3, 267-318.
[6] Backster R., Exactly soved models in statistical mechanics", M.: Mir, 1985. – 488 p.
[7] Belokon V.I., Nefedev K.V., "Magnetic ordering in 1D and 2D models of finite number Ising spins systems", Russian Physics Journal, №3/2, 2010, pp. 19-23.
[8] Kittel Ch., "Statistical termodynamics". M.: 1997, 336 p.
[9] Brian W. Kernighan, Dennis M. Ritchie, "C Programming Language", Published by Prentice-Hall in 1988, ISBN 0-13-110362-81988.
[10] John Benzi; M. Damodaran (2007). "Parallel Three Dimensional Direct Simulation Monte Carlo for Simulating Micro Flows". Parallel Computational Fluid Dynamics 2007: Implementations and Experiences on Large Scale and Grid Computing. Parallel Computational Fluid Dynamics. Springer. p. 95. Retrieved 2013-03-21.