International Journal of Networking and Computing – www.ijnc.org
ISSN 2185-2839 (print) ISSN 2185-2847 (online)
Volume 6, Number 2, pages 195–211, July 2016

Accelerator Chip for Ground-state Searches of Ising Model with Asynchronous Random Pulse Distribution

Masato Hayashi, Masanao Yamaoka, Chihiro Yoshimura, Takuya Okuyama
Center for Exploratory Research, Hitachi, Ltd.,
1-280, Higashi-koigakubo, Kokubunji, Tokyo, Japan

Hidetaka Aoki
Center for Technology Innovation - Information and Telecommunications, Hitachi, Ltd.,
292, Yoshida, Totsuka, Yokohama, Kanagawa, Japan

and

Hiroyuki Mizuno
Management Planning Office, Hitachi, Ltd.,
1-6-1, Maruno-uchi, Chiyoda, Tokyo, Japan

Received: February 15, 2016
Revised: May 4, 2016
Accepted: July 5, 2016
Communicated by Shuichi Ichikawa

Abstract

We propose a static random access memory based complementary metal-oxide semiconductor LSI chip that accelerates ground-state searches of an Ising model. Escaping local minima is a key feature in creating such a chip. We describe a method for escaping the local minima by asynchronously distributing random pulses. The random pulses are input from outside the chip and propagated through two asynchronous paths. In an experiment using a prototype of our chip, our method achieved the same solution accuracy as the conventional method. The solution accuracy is further improved by dividing the random pulse distribution paths and increasing the number of pseudo random number generators.

Keywords: Ising model, ground-state search, random pulse, accelerator

1 Introduction

Solving combinatorial optimization problems efficiently has become more important in recent years to improve the operation of infrastructures such like trasnportation systems. For example, in the area of traffic systems, traffic jams can be modeled by combinatorial optimization problems such as max-flow or shortest path problems. In logistics systems, total moving cost can be modeled by a traveling salesman problem or its variants. However, many combinatorial optimization problems are computationally intractable. Therefore, an efficient method is needed to solve large-scale combinatorial optimization problems appearing in social systems.

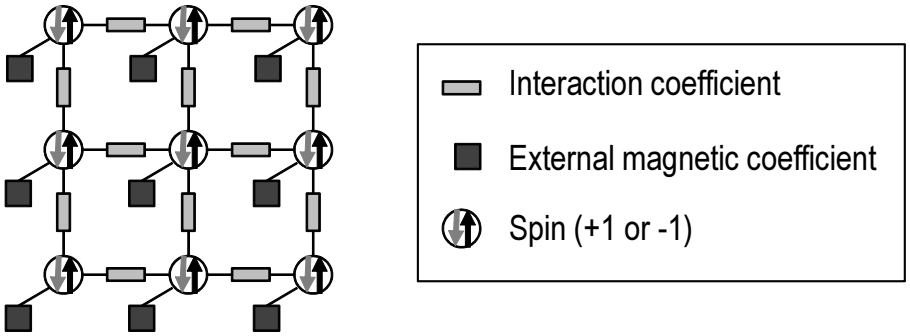


Figure 1: An example of 2D-lattice Ising model.

The conventional von-Neumann computer architecture has a performance limitation. Over the past few decades, the performance improvement of computers has been mainly achieved by the scaling of semiconductors. One of the major effects of this scaling is continuous improvement of the clock frequency. However, improvement in the clock frequency stopped prior to 2010 due to high power consumption. Thus, instead of clock frequency, the number of CPU cores has been increased to enable higher performance. However, it is difficult to parallelize code that was originally written sequentially for using many processing cores in CPUs. Thus, new algorithm that exploits higher parallelism is needed to achieve higher performance.

We previously proposed the concept of spatial computation to break this limitation and achieve higher computer performance [19]. In spatial computation, the problems to be solved are described by spatially distributed parameters. By doing so, we aim to exploit parallelism without restricting synchronization and the sequential part of the code. We further define new hardware specialized to process such spatial representation of the problems to enable higher performance than that of versatile computers.

We focused on the Ising model to describe the problem spatially. The Ising model was originally developed to describe the behavior of magnetic objects in statistical mechanics. As shown in Figure 1, the model consists of spins that take the value +1 or -1, interaction coefficients that describe the strength of the interaction between two spins, and external magnetic coefficients that describe the strength of external magnetic force at each spin. The problem in finding the combination of spin values that minimizes the energy of the Ising model is called the ground-state search problem, which is an instance of the combinatorial optimization problem. Therefore, the calculation requires two steps in spatial computation. The first step involves converting the problem to be solved into an equivalent ground-state search problem of the Ising model. Then, the ground-state search is executed, and its results correspond to the solution of the original problem.

We previously proposed a semiconductor chip specialized to solve the ground-state search problem of the Ising model [18]. Semiconductor chips are relatively easy to mass produce and to increase in size. Also, they can be easily integrated in versatile computers. The implementation of such hardware with semiconductors increases the profits of manufacturers.

For sparse Ising models, multiple spin values can be calculated in parallel during ground-state searches. We provide a simple compute unit (spin unit) for each spin in the Ising model so that our semiconductor chip can fully use parallelism. However, local minima need to be avoided because many such minima exist in the solution space of the ground-state search problem. In this situation, modifying the current solution by using random numbers is common way to avoid the local minima. The amount of random numbers that can be provided in each cycle limits the number of spin units that can operate in parallel because each spin unit requires a random number in every cycle. However, many pseudo random number generators (PRNGs) require a large area, reducing the number of spins integrated in one chip. Therefore, a method of providing random numbers with minimum hardware overhead is required.

This paper is organized as follows. In Section 2, we give an overview of the Ising model and

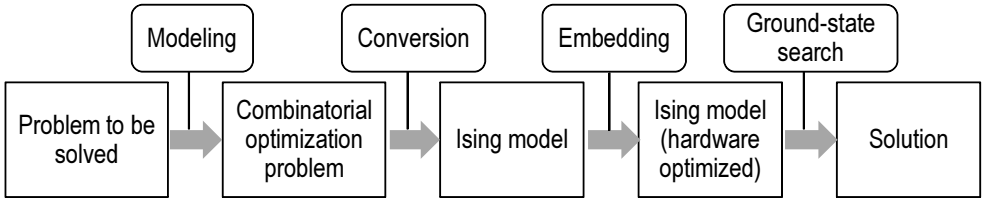


Figure 2: Computation flow to solve problems using Ising chip.

several method to solve the ground-state search problem of Ising model. In Section 3, we describe the computation flow to solve the combinatorial optimization problems using our proposed accelerator chip. We describe the architecture of our accelerator chip in Section 4 and present method to escape the local minima with minimal hardware overhead in Section 5. In Section 6, we explain the measurement results of a prototype of our chip and discuss its behavior. We describe related studies in Section 7 and conclude the paper in Section 8.

2 Ising model

The Ising model was developed by Wilhelm Lenz to model the behavior of magnetic spins [4]. It consists of an array of spins. Each spin (σ_i) takes a binary value +1 or -1. Interactions occur between two spins, σ_i and σ_j, and their strength is denoted as J_{ij}. External magnetic fields are produced at each σ_i, and their strength is denoted as h_i. The energy function (Hamiltonian) of the Ising model is calculated using the following equation.

$$H(\sigma) = \sum_{i<j} J_{ij}\sigma_i\sigma_j - \sum_i h_i\sigma_i \tag{1}$$

where σ is a set of all spin values. When the value of J_{ij} is zero, there is no interaction between σ_i and σ_j. Interaction coefficient values determine the topology of the Ising model. For example, the Ising model forms a complete graph when all coefficients have non zero values. In contrast, if most of coefficient values are zero, it may form a sparse topology similar to a lattice. For the spin configuration, the energy value of the Ising model minimum is called the "ground state," and the operation for finding the ground state is called the "ground-state search."

The computational complexity of the ground-state search problem is different from the topology of the Ising model. The ground-state search problem for the Ising model, which its topology is a planar graph and has no external magnetic field, can be reduced to a perfect matching problem of graphs [1]. This problem can be solved using Edmonds' algorithm in polynomial time $O(N^3)$ where N is the number of vertices in the graph [8]. Otherwise, it has been shown to be an NP-hard problem when the Ising model is a non-planar graph or has external magnetic coefficients [1]. Current algorithms for ground-state search of the Ising model are classified into two categories. One is to find an exact solution and the other is to find approximate solutions. The branch-and-cut algorithm [7] is former. Although these algorithms can find an exact solution for moderate-sized models, it requires long execution time, which substantially differs according to the input problem. An example of the latter is metaheuristics such as simulated annealing (SA) [12] or genetic algorithms. The Ising chip is categorized as the latter, and is aimed to quickly achieve approximate results. Note that we use the term "ground-state" including approximate one. We use the energy value of the approximate ground-state for comparing the solution accuracies of the approximate method. In other words, lower energy means higher accuracy.

3 Computation flow using Ising chip

In conventional computers, once the problem for imroving the operation of infrastructure systems is modeled by combinatorial optimization problems, it is usually solved directly using specialized

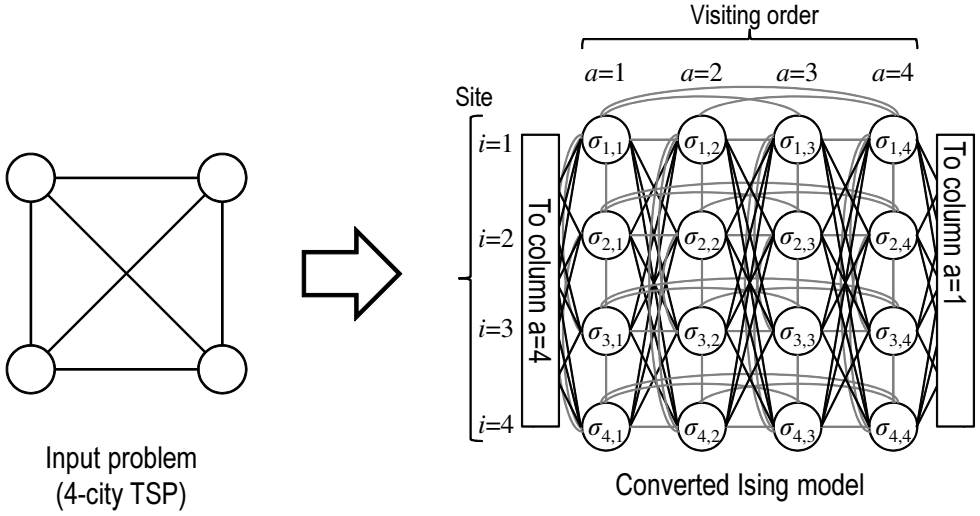


Figure 3: Conversion example of 4-city TSP in Ising model

algorithms or metaheuristics. On the other hand, in Ising computing, the problems are solved according to the flow shown in Figure 2. Each step is explained below.

Modeling

In the modeling step, the problem to be solved is modeled by the combinatorial optimization problem. For example, reduction in the total moving cost of trucks in the field of logistics can be formulated as a traveling salesperson problem (TSP), which is one of the most well-known combinatorial optimization problems. This step is the same as the computing flow using conventional computers.

Conversion

The combinatorial optimization problem, which is the result of modeling step, should be converted to the Ising model. The conversion problem consists of two parts. One is to represent the solution of the original problem by the combination of binary spins and the other is to determine the values of interaction coefficients and external magnetic coefficients so that the spin values at the ground-state correspond to the solution of the original problem. Some conversion methods are known in the literature [15].

We show the conversion method for a TSP as an example. The objective with a TSP is to find the shortest route to visit all cities exactly once from the input list of cities and the distance between each pair of cities. This invloves N^2 spins for the N-city TSP. Note that each spin value takes 0 or 1. Figure 3 shows the conversion results for a 4-city TSP problem. As shown in the figure, N^2 spins are placed in a square. The number of cities is represented by the position of the spin with 1 in a column. For example, $\{s_{1,1}, s_{2,1}, s_{3,1}, s_{4,1}\} = \{0, 1, 0, 0\}$ represents city 2 because the second spin in a column takes 1. Similarly, visiting a city is represented by the position of the spin with 1 in a row. The permutation of the cities, which is the solution to this TSP, can be achieved by reading the value of each column in turn. To achieve a feasible solution of this TSP, exactly one spin in each row and column must take 1 because each city must be visited only once. The energy function of the Ising model that solves a TSP based on the above discussion is shown with the following equation:

$$H(s) = \sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{a=1}^{N} D_{ij} s_{ia} s_{j,a+1 mod N} + \gamma \left[\sum_{a=1}^{N} \left(1 - \sum_{i=1}^{N} s_{ia} \right)^2 + \sum_{i=1}^{N} \left(1 - \sum_{a=1}^{N} s_{ai} \right)^2 \right] \quad (2)$$

where D_{ij} are constants that represent the distance between city i and city j, s are 0-1 variables and γ is a constant greater than zero. The first term represents the total distance of the route. The D_{ij} is added only when the j-th city is visited immediately after the i-th city is visited. The second term represents the constraint condition of a TSP. Generally, equality constraints $\sum_i s_i = a$ can be added to the energy function as the term $(a - \sum_i s_i)^2$ because the value of the term takes 0 only when $\sum_i s_i = a$; otherwise takes a greater value. Therefore, it can be used as a penalty term that increases the energy value when the constraint is not satisfied. The γ is a parameter that controls the effect of constraint.

Embedding

The converted Ising model may have an arbitrary topology and wide range of coefficient values. Although hardware that can handle them may be implemented, there are two issues. One is the increase in circuit area. When the converted model has a very dense topology, such as complete graphs, the processing amount of each spin will be increased because each spin must calculate the force from all adjacent spins. This lead to an increase in circuit area. In addition, a large amount of memory is required to record a wide range of coefficient values. This also leads to an increase in circuit area. The other problem is degradation in parallelism. In general, multiple spins can be updated in parallel while performing the ground-state searches of the Ising model. However, adjacent spins cannot be updated in parallel because both update conflicts and spin values oscillate. In complete graphs, each spin is adjacent to all other spins in the graph; therefore, only one spin can be updated at a time. This is sequential processing. Therefore, little or no parallelism is available in a dense topology and almost no increase in speed can be expected from the hardware.

Therefore, it is important to keep the topology of the Ising model represented by the hardware sparse and run as many spins in parallel as possible. To use such hardware, another method for transforming the converted Ising model into an optimized one into specific hardware is desired. We call this transformation "embedding". Embedding consists of the processes. One is transforming the topology of the input Ising model to a sparser one. For this purpose, "minor-embedding" techniques have been is proposed [5, 6]. They convert the input Ising model into a sparser one while maintaining the spin values at the ground-state. They divide the spin with high degree into multiple spins and set the coefficient values between divided spins so that they keep the same value as possible. Although some embedding methods exists, an efficient method that uses less spins and can be executed in less time is needed. The other process is rounding the range of coefficient values without degrading solution accuracy. To the best of our knowledge, this method has not yet been established.

At this time, the lack of an embedding method is major a barrier to solve real-world problems using the Ising chip, which will be our research topic in the near future as well as improvement in the architecture of the accelerator chip for more efficient ground-state searches.

Ground-state search

In the final step, converted and embedded Ising model is input to the hardware. Finally, the solution of the original problem is restored from the spin values from the results of the ground-state search.

4 Accelerator Chip for Ground-state Searches of Ising Model

4.1 Overview of Ising chip architecture

The Ising chip is a static random access memory (SRAM)-based chip that executes the ground-state search of the Ising model. In this study, we focused on validating our concept of spatial computing,i.e. to exploit sufficient parallelism by converting the combinatorial optimization problems into the ground-state searches of the Ising model. Therefore, we designed an Ising chip to integrate more spins into a chip by making the circuit correspond to a spin be as simple as possible.

Our chip has mainly two functions; to record the coefficient and spin values of the Ising model and to the perform ground-state search. Ground-state search is made possible by a simple interaction

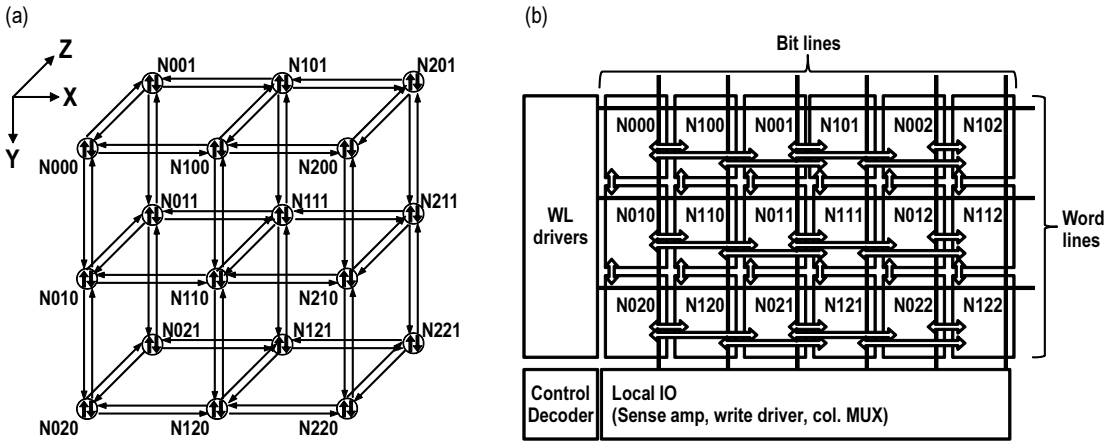


Figure 4: (a) Logical topology that can be represented with proposed chip. Each digit following letter 'N' means coordinate of spin.(b) Block diagram in chip plane.

operation at each spin. Figure 4 (a) shows the topology of the Ising model that can be represented with our chip. As shown in the figure, spins are connected to each other to form a 3D-lattice with the size of the Z-axis being 2.

The main part of our Ising chip is an array of spin units. A spin unit is an element circuit that mimics a single spin in the Ising model. A spin unit has SRAM cells to store values and the interaction circuit to perform the ground-state search. The details of spin units are described in Section 4.2. Spin units are placed on the chip plane, as shown in Figure 4 (b). Spins along the Z-axis of the Ising model are embedded into the X-axis of the spin array on the chip plane. To read or write the coefficient and spin values, the chip has a standard SRAM interface, which consists of a word line, bit line, word line driver, and input/output (I/O) circuit.

The interaction operation is executed synchronously with the interaction clocks. At each clock, multiple spin units can execute the operation in parallel. However, adjacent spin units cause conflict of interaction because both spin units try to determine their next value using each others old spin values. Therefore, spin units are divided into non-adjacent groups, and only one group executes the interaction operation at one clock. Although at least two groups are sufficient for 3D-lattice topology to avoid conflict, it may consume too much power to execute interaction by half of all spin units at one clock. For this reason, there are eight groups in our chip.

4.2 Spin unit

The structure of a spin unit is shown in Figure 5. It mainly consists of memory cells and an interaction circuit. Memory cells in the left part of the figure store the interaction coefficients with their adjacent spins, external magnetic coefficient, and spin value. The interaction circuit in the right part of the figure calculates the next state of the spin value. As described in Section 4.1, spin units are connected to form a 3D lattice with the size of the Z-axis being 2. Thus, each spin unit has five adjacent spin units, that is, left, right, up, and down directions in the XY plane and back or front direction along the Z-axis. A spin value takes +1 or -1 and is mapped to the 1 or 0 state of a SRAM cell, respectively. A spin value is stored in the cell called N. The interaction and external magnetic coefficient values are stored in the 2-bit SRAM cell called Ix0 and Ix1 (x is S, U, L, R, D, or F). Each coefficient takes +1, 0, or -1. The term Ix1 represents the value +1 or -1 and Ix0 enables the bit and coefficient values to be 0, regardless of Ix1 when Ix0 is 0.

The interaction operation performed by the interaction circuit calculates the next spin value that minimizes the local energy. The local energy around spin σ_i is calculated using the following

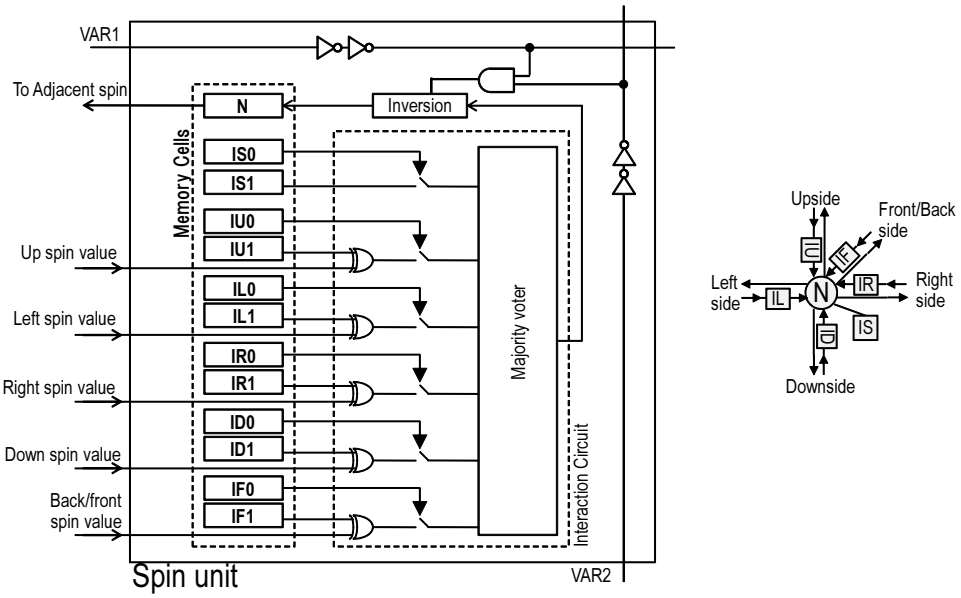


Figure 5: Structure of single spin unit and its connection with neighbor spin unit.

equation.

$$H_{local}(\sigma_i) = \sum_{j \in adj(i)} J_{ij}\sigma_i\sigma_j - h_i \tag{3}$$

where $adj(i)$ is a set of indices of all spins adjacent to i. Since the spin value is either +1 or -1, the local energy can be minimized by setting the spin values by the following rule:

$$\sigma_{i,t+1} = \begin{cases} +1 & \text{if } F(i) > 0 \\ -1 & \text{if } F(i) < 0 \\ \text{Don't care} & \text{otherwise} \end{cases} \tag{4}$$

$$F(i) = \sum_{j \in adj(i)} J_{ij}\sigma_{j,t} - h_i$$

where $\sigma_{i,t}$ is the value of spin σ_i at time t. This rule can be met using an adder and multiplier. However, this requires a large circuit. Instead, a simpler circuit can be used because the coefficient values are restricted to +1, 0, or -1 in this case. The product of the adjacent spin value and coefficient values can be calculated by the exclusive-or gate. Then, the sign of the sum can be determined by a majority of voter circuits. We used an analog circuit to further reduce the size of the spin unit [18]. It represents the force to pull the spin value to +1 or -1 by the current and detects the difference in the current using the sense amplifier.

5 Asynchronous Random Pulse Distribution Path to Escape Local Minima

As described in Section 4, the interaction operation determines the spin value to lower the local energy around the target spin. That is, the energy of the whole Ising model cannot be increased by the interaction operation. However, as shown in Figure 6, many local minima exist in the energy landscape of the Ising model. When only using the interaction operation, the nearest local minima from the initial configuration can be reached. Accurate solutions cannot be acquired in this situation because the solution space is not searched sufficiently. Therefore, escaping the local minima is essential in achieving higher accuracy.

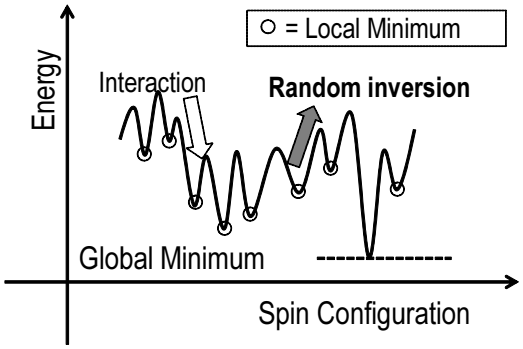


Figure 6: Schematic energy landscape of Ising model

Generally, some sort of randomness can be used to escape the local minima. Simulated annealing (SA), a widely used metaheuristic for optimization problems, is an example. It combines local search with an escape local optimum solution using randomness. In general, SA consists of the following steps. A neighbor solution of the current solution is generated. Then, the value of the evaluation function is calculated for both the current and neighbor solution. The neighbor solution is accepted using the following probability. Note that minimization of the evaluation function is assumed.

$$p_{accept} = \begin{cases} 1 & \text{if } E(n) < E(c) \\ \exp(-\frac{E(n)-E(c)}{T}) & \text{otherwise} \end{cases} \quad (5)$$

where E is the evaluation function, c is the current solution, n is the neighbor solution, and T is temperature. If the neighbor solution is accepted, it becomes the new current solution. Otherwise, the neighbor solution will be discarded. "Equation (5)" shows that the better solution is always accepted. On the other hand, a worse solution is also accepted probabilistically according to the temperature and difference in the evaluation function. A worse solution is more frequently accepted at higher temperature.

The behavior of our Ising chip is somewhat similar to SA. For the Ising model, the evaluation function is the energy function of the Ising model in Equation (1). An exact hardware implementation of SA for the ground-state search problem must calculate the difference in energy and exponential functions to determine the acceptance probability. This requires a complex spin unit and decreases the density of spin units. Therefore, we developed a simplified method to escape local minima. With our method, each spin unit simply inverts its value according to a certain probability, regardless of the difference in the energy value. From the view point of a spin unit, just a 1-bit random value is needed to determine whether to invert its spin value. This method significantly reduces the complexity of a single spin unit. An efficient method for providing a 1-bit random value for all spins is thus needed. Of course, a naive implementation in which all spin units contain a PRNG also considerably increases the circuit size of the spin unit. Instead, we provide randomness for all spin units with a minimum number of PRNGs using an asynchronous path.

Our Ising chip uses random pulse sequences that are input from outside it. Our Ising chip has two zigzag paths, horizontal and vertical, to distribute a random pulse, as shown in Figure 7 (a). Both types go through all spin units. Independent random pulse sequences are input for each path. Each path consists of a long inverter chain, shown in the upper part of Figure 7 (b). Random pulses propagate through the paths asynchronously with an interaction clock, as shown in the lower part of the figure. Each spin unit escapes the local minima by inverting its spin value when the values of the random pulses from both paths are "1." The hardware cost of the paths is low because the value of the random pulses is only 1 bit.

The mark ratio of the random pulse sequence, the probability indicating how often the value "1" appears in the sequence, is an important parameter. It corresponds to the probability to invert the spin value. In other words, it shows how often each spin unit escapes the local minima. During the ground-state search, the mark ratio is set high to easily escape the local minima at the beginning of

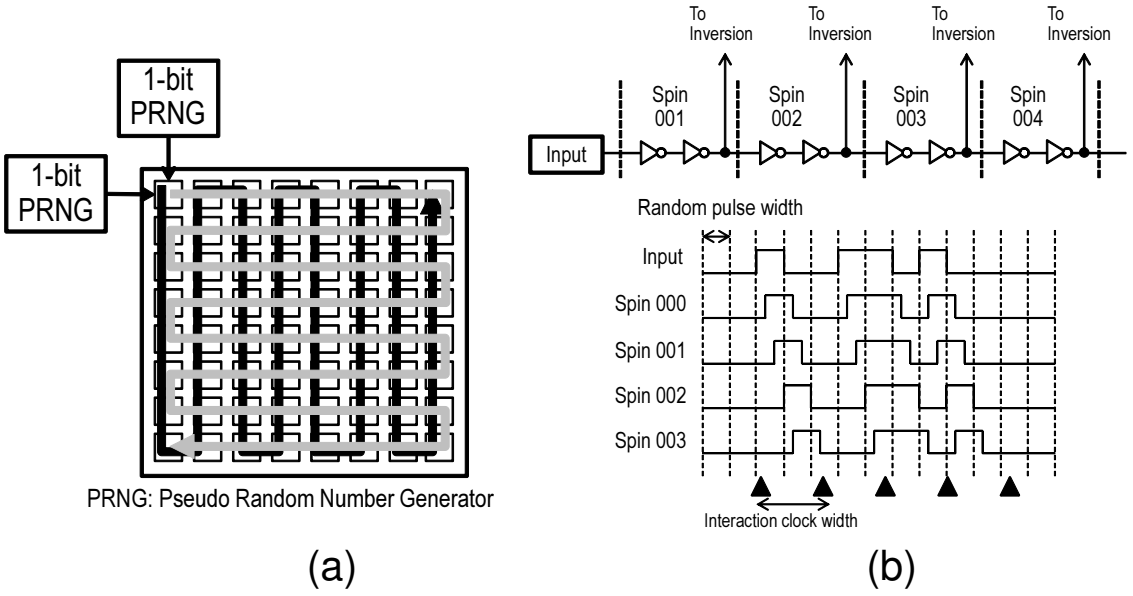


Figure 7: (a) Paths to propagate random pulses input from outside chip. (b) Structure of single path and timing chart of propagating pulses.

the calculation. As the calculation progresses, the mark ratio is lowered to settle the ground-state. In summary, the energy of the Ising model is lowered by the interaction operations that lower the local energy and random inversion of the spin values that escape the local minima at the same time. In our Ising chip, the mark ratio is a similar parameter to temperature in SA.

The interaction clock and random pulse frequencies should be set appropriately to achieve higher accuracy. For example, a higher interaction clock frequency is advantageous for increasing the speed of the ground-state search. However, random pulses input from outside the chip will propagate at a constant speed. Therefore, an interaction operation may be executed before the previous random pulses sufficiently go through the spins. In this situation, the value of the random pulses at each spin tend to remain the same, which may degrade solution accuracy. However, the random pulse frequency does not affect the speed of the ground-state search because the time required to calculate only depends on the number of interactions and interaction clock frequency. A lower frequency of random pulse generation may make the design of the PRNG easier and may result in lower power consumption. However, if the random pulse frequency is too low, the same value will be input across many spins. This may degrade solution accuracy because the value of the random pulses is no longer independent to that of the neighboring spins.

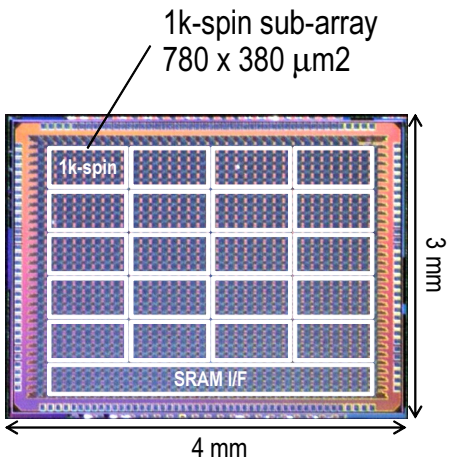
6 Experiment and Discussion

6.1 Experimental Conditions

We fabricated a prototype of our Ising chip described in Sections 4 and 5 using a TSMC 65-nm process. The specifications of the prototype chip are listed in Figure 8(b). The area of the chip is $12mm^2(3 \times 4mm)$. It integrates 20,480 ($128 \times 80 \times 2$) spin units. The maximum speed of the SRAM I/F and interaction is 100 MHz. A photograph of the chip is shown in Figure 8 (a).

The maximum cut problem, which is an NP-complete combinatorial optimization problem, was used as the benchmark. The maximum cut problem can be mapped into the ground-state search of the Ising model by multiplying -1 by all weights of the edges. The size and shape of the problem is a $128 \times 80 \times 2$ 3D-lattice, which is the same as that of our prototype chip. The weights were uniformly chosen at random from +1 or -1. For comparing the solution accuracies, the SG3 algorithm [11],

(a)



(b)

Items	Value
Number of spins	20,480 (128x80x2)
Process	65nm
Number of SRAM cells	260k bits
Interaction clock frequency	100 MHz (max.)
Memory clock frequency	100 MHz (max.)
Chip area	12 mm^2 (3 mm x 4 mm)
Process	65 nm
Operating voltage	1.1V
Operating current of core circuits (1.1V) (Do not include I/O)	Write: 2.0 mA Read: 6.0 mA Interaction: 44.6 mA

Figure 8: (a) Photograph of prototype chip. (b) Specifications of prototype chip.

Table 1: Measured energy of Ising model achieved with prototype chip

		Interaction Clock Frequency (MHz)			
		12.5	25	50	100
Random Pulse Frequency (MHz)	12.5	-29,478	-28,573	-28,355	-28,332
	25	-30,396	-29,491	-28,432	-28,439
	50	-30,833	-30,190	-29,064	-28,610
	100	-30,603	-30,206	-29,176	-28,745

which is a state-of-the-art heuristic algorithm for the maximum cut problem, was used. It was executed on a standard PC with Intel® Core i5® 1.87-GHz CPU and 8-GB memory. As described in Section 5, the combination of interaction clock and random pulse frequencies affects solution accuracy. To evaluate this effect, they were varied between 12.5 and 100 MHz respectively. The number of interaction clocks during the calculation was set to 100,000. The mark ratio of the random pulses at clock t was set using the following equation:

$$p(t) = p_{initial} \cdot \exp(\log \frac{p_{final}/p_{initial}}{N}) \tag{6}$$

where $p_{initial}$ is the mark ratio of random pulses at the start of the interaction, p_{final} is that of random pulses at the end of interaction, and N is the total number of interactions. In this evaluation, $p_{initial}$ was set to 0.75 and p_{final} was set to 0.01. The mark ratio was set to 0 at the last 1000 clocks of interaction to ensure the solution was at least one of the local minima. Note that each spin unit determines whether to invert the spin value according to the logical AND of two random pulses. Therefore, the expected probability of the invert spin value was approximately the squared value of the mark ratio. For the prototype chip, we solved the same problem 100 times, and average energy values were used because the calculation results slightly differed in every trial due to randomness.

Table 2: Speed comparison between conventional computer and the prototype chip

Interaction clock frequency (MHz)	Execution Time (msec)					Increase in speed
	Ising chip				SG3	
	Write	Interaction	Read	Total		
12.5	0.166	8	0.013	8.179	15.4	1.88
25		4		4.179		3.69
50		2		2.179		7.07
100		1		1.179		13.06

6.2 Actual Chip Results

The measured energies of the Ising model are listed in Table 1. The energy achieved with SG3 was -30,254. Note that lower energy indicates higher accuracy. An accuracy higher than that with SG3 was achieved with the prototype chip when the interaction clock frequency was 12.5 MHz and random pulse frequency was higher than or equal to 25 MHz. Our method of using just two PRNGs can achieve a sufficiently accurate solution for the ground-state search of the Ising model. However, the solution accuracy was lower when the interaction clock frequency was higher than the random pulse frequency. When the interaction clock and random pulse frequencies were the same, the solution accuracy decreased as the frequency increased. This phenomenon is further discussed in Section 6.3.

Table 2 lists the execution times for solving the problem with our prototype Ising chip and SG3. The time for the chip consists of three parts. The first part is to write coefficient values and initialize the spin value (write). The second part is to execute the interaction (interaction), and the last part is to read the resulting spin value (read). The execution time for each part depends on the number of clocks spent on that part because our prototype chip can execute a read and write operation for a single address and an interaction operation in one clock. Write and read operations do not affect solution accuracy and are always executed in 100 MHz independent of the interaction clock frequency. The difference in time between write and read is due to the difference in the amount of memory to write or read. All memory cells in the chip are written once in the write part. On the other hand, only parts of memory cells that correspond to spin values are read in the read part. Therefore, the read part requires much less time than the write part.

As shown in the table, the prototype chip was 1.88 to 13.06 times faster than with SG3 for the maximum cut problem with 3D-lattice topology. Although solution accuracy degrades at higher interaction clock frequency, higher speed than SG3 can be achieved by running many simple spin units in parallel. This is what the concept of spatial computing is expected to do.

There are two major factors that affect the speed of ground-state searches other than the interaction clock frequency. One is the number of interactions during the ground-state search. There are trade-offs between the number of interactions and solution accuracy. More accurate solutions tend to be achieved when the mark ratio is lowered more slowly because a wider area of solution space is searched. This requires many interactions until the mark ratio is lowered sufficiently. Therefore, the time required for the ground-state searches is increased. Although the number of interactions was fixed in this evaluation, it can be reduced for applications that do not require high solution accuracy. For such applications, execution time for the ground-state search can be reduced by reducing the number of interactions.

The other factor is the number of spins in the prototype chip. The SG3 algorithm cannot be accelerated by the increase in the number of CPU cores because it is a sequential algorithm. Therefore, the execution time of SG3 increases as the problem size increases. On the other hand, the execution time for the ground-state searches in the prototype chip increases little as the number of spin units in the chip increases because each spin unit runs in parallel. This is also true for the execution time for solving the maximum cut problems. Therefore, the relative performance of the prototype chip against SG3 can be improved by integrating more spin units into the chip. As discussed in Section 4, an Ising chip has an array-like structure of basic elements called spin

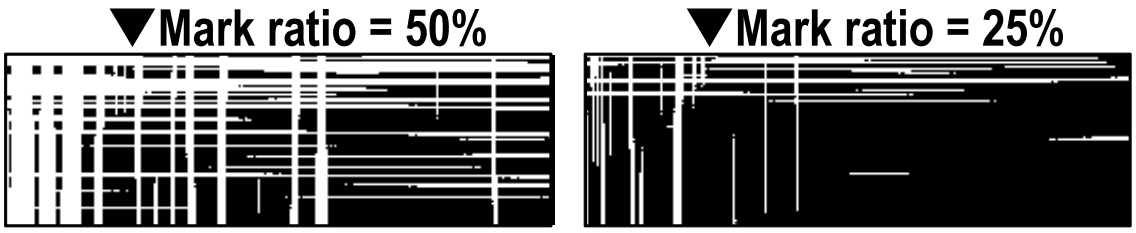


Figure 9: Observed random pulses in prototype chip

units; therefore, a larger chip can be easily fabricated with the same architecture. In addition, the prototype chip is relatively small and fabricated with an old process. Therefore, there is still room for fabricating larger chips. We estimate up to 1 million spin units can be integrated in a single chip using 28-nm process [18]. For such larger chips, relative increase in speed of CPUs is expected to be higher than that of the prototype Ising chip.

From these resulsts, we showed the concept of spatial computing that exploit parallelism by using many simple spin units. Unfortunately, our prototype Ising chip cannot solve more complex problems than maximum-cut as shown in Section 3 because its range of the coefficient values are too narrow. Extending the architecture of our Ising chip to solve wide range of combinatorial optimization problems is our future work.

6.3 Behavior of random pulse propagation

The behavior of random pulses in a chip may lower accuracy because the accuracy is affected by the combination of interaction and random pulse frequencies. We first observed the behavior of random pulse propagation through the prototype chip to inspect the cause of lower accuracy. We used an indirect method of observing the value of the random pulse at each spin unit because the prototype chip had no such function. We set all interaction coefficients to +1 and the initial spin value to +1. In this situation, all spin values remained +1 during the interaction if no random pulses occurred. Thus, the position of random pulses "1" could be observed by finding spin value -1. The observation results are shown in Figure 9. The interaction clock and random pulse frequencies were 100 MHz. The black points indicate that both random pulses were 1 (spin value is inverted at the spin) and the white points indicate at least one random pulse was 0 (spin value is not inverted at the spin). The mark ratio was 25 and 50%. The expected flip rate was 6.25 and 25%, respectively. The black points appeared more frequently than expected in both cases. In particular, the black points were dominant in the bottom right corner of the chip. These results show that the actual flip rate of the spins was significantly higher than expected. This may have degraded solution accuracy.

Figure 10 (a) shows the position of random pulses at each clock from pulse input. The pulse position on the vertical axis is the number of spins counted from the start of the path to the tail of random pulse along the random pulse path. That is, position 0 means the tail of a random pulse is at the start of the path, and position 20,480 (total number of spins) means the pulse reached the end of the path. The figure shows that a random pulse propagates at a constant speed in both paths. The difference in propagation speed between paths is due to the difference in the physical length of the paths in the prototype chip. The figure shows that there is little fluctuation in the propagation speed of random pulses.

On the other hand, Figure 10 (b) shows the variation in the length of a single pulse during propagation in the prototype chip. The pulse length means how many contiguous spins the pulse value "1" spans. This indicates that the length of the single pulse increased and finally reached 7 to 10 times the initial length. This result shows that the mark ratio near the end of paths was significantly higher than expected. Therefore, a portion of spins likely invert their spin values at the end of ground-state search. In the prototype chip, random inversion of the spin value always increases the energy because the interaction operation determines the spin value to reduce energy.

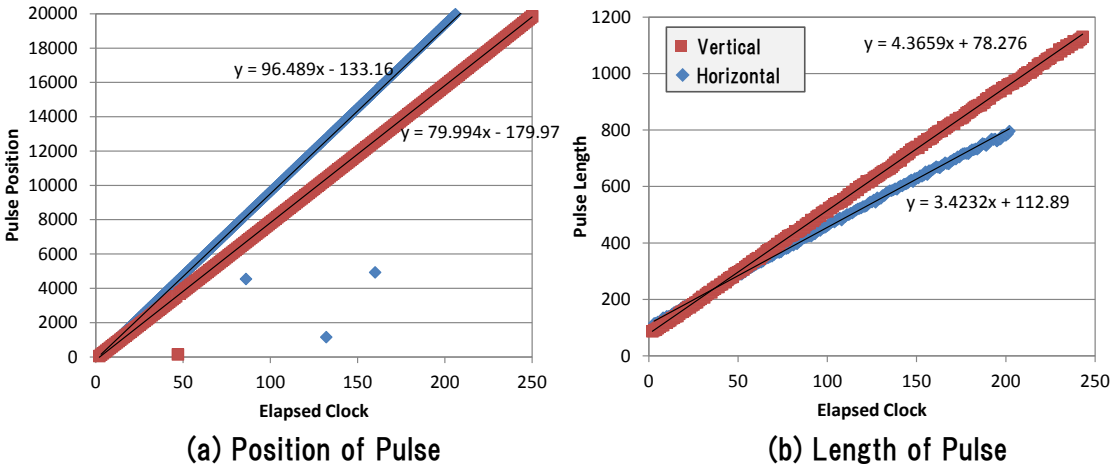


Figure 10: Behavior of single random pulse in prototype chip. (a) Position of the pulse (b) Length of pulse

As a result, it degrades the accuracy that the random pulse length becomes longer.

As shown in Figure 7, two inverters are contained in a spin unit as the path for random pulses. This pair of inverters can be simplified as an element that delays an input signal. When the delay time of the inverter pair is always constant for any input change, the form of the random pulse can never be changed during propagation. Therefore, the difference in delay time between "rise" and "fall" causes a variation in the length of random pulses. "Rise" means that the input of the inverter pair changes from 0 to 1, and "fall" means vice versa. Although the difference arises from the physical implementation of the inverter pair, we consider the effect of the difference in delay time on the accuracy of ground-state search.

First, we consider the relation between the difference in delay time and variation in pulse length. Let R be the delay time for rise, F be the delay time for fall, and T be the clock period. We assume a sufficiently long random pulse path and only one pulse. The head of the pulse advances $\frac{T}{R}$ spins per clock and the tail of the pulse advances $\frac{T}{F}$ spins per clock. Therefore, the pulse length varies by $\frac{T}{R} - \frac{T}{F}$ per clock. The length of the pulse with value 1 becomes longer when R $<$ F. Figure 10 (a) shows that the delay time is around 80 to 100 psecs. Figure 10 (b) shows that the pulse length becomes longer by 3 to 4 spins per clock. For example, if the delay time for rise is assumed to be 100 psecs, that for fall will be 104 psecs.

The effect of the difference in delay time on the accuracy of the ground-state search was evaluated using the behavior level simulator of the prototype chip. We measured the energy value achieved by ground-state search with a certain set of delay times. The delay time for rise was fixed to 100 psecs and that for fall varied between 100 to 110 psecs. For this evaluation, the interaction and random pulse frequencies were both set to 100 MHz. Figure 11 shows the measurement results. The vertical axis shows the energy relative to the ideal case in which the delay times for rise and fall are completely the same. The results show that accuracy rapidly worsened as the difference in delay time increased. When the delay time for fall was around 110 psecs, the solution accuracy was almost the same in the case in which no random pulses were input (corresponding energy shown with black line in the Figure 11). This shows that a large difference in delay time completely disables the prototype chip from escaping the local minima.

6.4 Dividing random pulse paths

The problem of varying the length of pulses can be resolved by carefully designing inverter pairs and making the difference in delay time approximate to 0. However, highly precise design is required to achieve high accuracy because a small difference substantially degrades the solution accuracy.

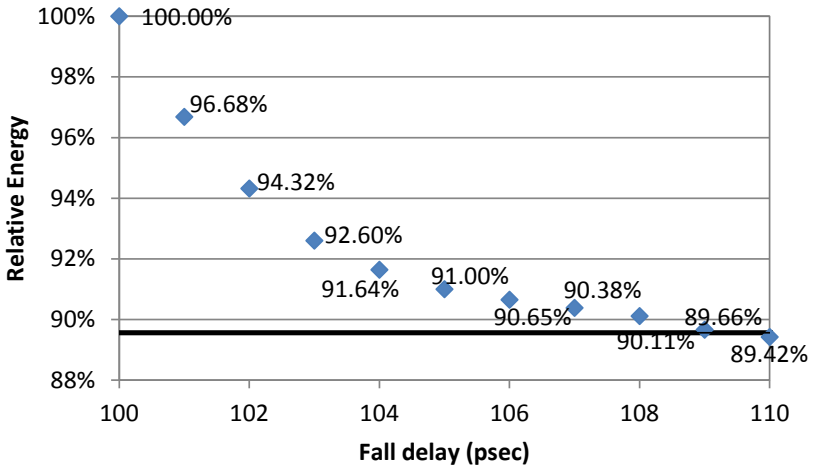


Figure 11: Relative solution accuracy for various delay times of inverter pairs

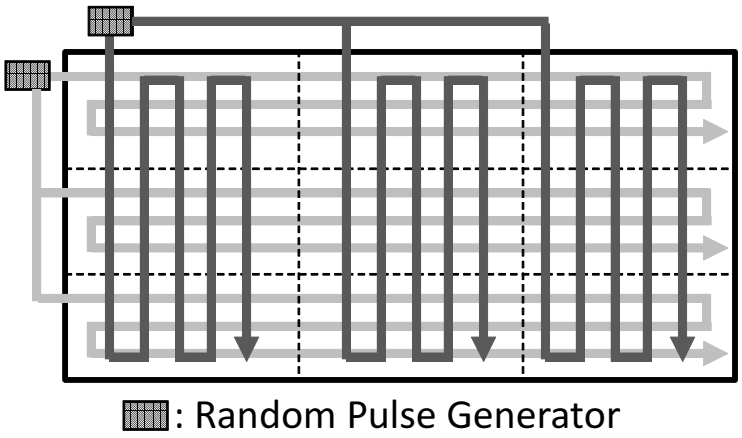


Figure 12: Divided random pulse paths

Additionally, faster interaction and random pulse frequency for future increase in speed of the chip shortens the single pulse length . This requires even smaller difference in delay time. Therefore, another method of improving accuracy that is less sensitive to the difference in delay time is required. The variation in random pulse length increases as the pulses pass many spins. Thus, a shorter path helps improve accuracy.

We designed a configuration that divides the paths of random pulses to improve solution accuracy. In this configuration, the spins lying in a 2D plane are divided into different groups along the X- and Y-axes. A pair of PRNGs is shared with all groups to prevent an increase in the hardware cost. Figure 12 shows an example that divides a chip into 3×3 groups along the X-axis and Y-axis, respectively. The effect in solution accuracy was evaluated using the behavior level simulator of our Ising chip. The simulator takes variations in the length of pulses into account. The delay time for rise is set to 100 psecs and that for fall is set to 104 psec. The interaction clock frequency and random pulse frequency are assumed to be 100 MHz.

The results of the experiment are shown in Figure 13. The relative energy against that of undivided cases is plotted in the chart. The horizontal axis is the sum of the number of groups along the X- and Y-axes. Note that higher relative energy means better accuracy. The highest accuracy was achieved at 32 blocks (16 and 16) with an improved accuracy of 7.5%. However, a higher number of blocks worsened accuracy. The cause was that the same random pulses were input to adjacent

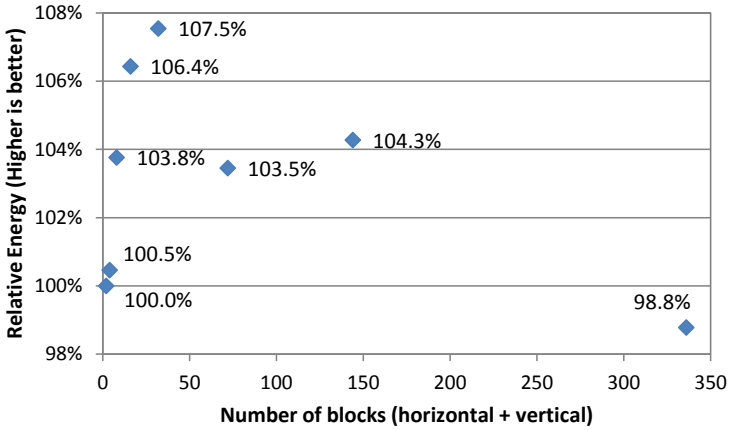


Figure 13: Solution accuracy for divided random pulse paths

spins because the same pair of PRNGs was shared. This result shows that the appropriate division of random pulse paths can improve solution accuracy without increasing hardware cost.

7 Related Work

Studies on accelerating the calculation of the Ising model using certain hardware can be categorized into two groups based on their objective. One is for Monte Carlo simulation of the spin system. The other is for ground-state search of the Ising model. The former includes GPU-based [3, 13, 17] and field-programmable gate array (FPGA)-based [2, 9, 14] implementations. Most GPU-based implementations use the NVIDIA®CUDA®language. Basically, an operation to update single spins is described as a single thread and mapped to each streaming processor core on the GPU. Also, efficient utilization of memory hierarchy is important to achieve higher performance because memory access is very costly in GPUs. A similar approach that divides the original Ising model with a 2D or 3D-lattice topology into sub-lattices is used in these implementations. Each sub-lattice is processed by the group of threads that shares fast memory on the GPU. These implementations have independent linear congruent RNPGs for each thread to provide randomness for updating a spin that is based on the metropolis algorithm.

A GPU can execute Monte Carlo simulation more effectively than a CPU due to high parallelism. However, it consumes a relatively high amount of power, around 100 W. Also, GPUs are basically designed for massive floating-point operation, which may be too rich for the Ising model with relatively narrow integer coefficients.

The latter implementation, FPGA-based implementation, basically stores coefficient and spin values into on-chip or off-chip and many processes read and update the spin values. Gilman [9] and Lin [14] implemented pipeline architectures on a single FPGA that integrate 256^3 and 1024^2 spins respectively. On the other hand, Belletti et al. [2] developed Janus, which was designed for larger systems. The entire system consists of multiple compute cores and hosts. Each core has a 4x4 2D grid of processing elements and I/O processor. This system is flexible in terms of the topology of the Ising model and behavior of updating rule. It offers higher flexibility and scalability in exchange of large hardware resource.

A specialized processing element for updating the spin value of the Ising model can be designed for an FPGA-based system. Therefore, relatively low power consumption can be expected. However, there is still room for further optimization because the ratio of elements in an FPGA, such as look up table for user-defined logic and flip-flops, are fixed, and this ratio does not always fit for implementing the desired circuit.

With some systems for Monte Carlo simulation, coefficient values of the Ising model are assume

to be fixed [3, 9, 14]. This is a critical difference between simulation purpose and ground-state search purpose because this assumption is not applicable for solving the optimization problems via the Ising model. For optimization, the original problem has to be represented by a combination of coefficient values and cannot be fixed.

On the other hand, hardware implementations that perform the ground-state search of the Ising model have been proposed [10, 16]. Johnson et al. [10] proposed D-Wave, which provides a quantum annealing machine using superconducting flux qubit. The National Institute of Informatics [16] proposed a coherent Ising machine that maps spin values to the phase of laser pulses in a ring cavity. Although these implementations may achieve higher accuracy or speed than conventional implementations, it seems that there are technical problems in increasing the number of spins in the system and downsizing of the entire system.

Therefore, application-specific integrated-circuit-based implementation has the advantage for accelerating optimization problems for social systems because it can be easily embedded in other systems or used in a power-restricted environment. We also proposed efficient architecture that integrates memory cells and interaction circuit for updating the spin value.

8 Conclusion

We proposed a CMOS chip that executes the ground-state search of the Ising model. The chip implements an efficient method to distribute random pulses asynchronously with at least two PRNGs. We fabricated a prototype chip using a 65-nm process and evaluated its solution accuracy. The results showed that our method of only using two PRNGs can achieve the same solution accuracy of that with an SG3 algorithm, which is a greedy algorithm executed on a conventional computer. In the prototype chip, the length of the random pulses increases as they go through the chip. Dividing the random pulse paths is effective in alleviating this problem. We also evaluated the effect of dividing the paths through simulation. The solution accuracy improved by 7.5% from the original configuration.

References

[1] Francisco Barahona. On the computational complexity of ising spin glass models. *Journal of Physics A: Mathematical and General*, 15(10):3241, 1982.

[2] Francesco Belletti, Maria Cotallo, Andrés Cruz, Luis Antonio Fernandez, Antonio Gordillo-Guerrero, Marco Guidetti, Andrea Maiorano, Filippo Mantovani, Enzo Marinari, Victor Martin-Mayor, et al. Janus: An fpga-based system for high-performance scientific computing. *Computing in Science & Engineering*, 11(1):48–58, 2009.

[3] Benjamin Block, Peter Virnau, and Tobias Preis. Multi-gpu accelerated multi-spin monte carlo simulations of the 2d ising model. *Computer Physics Communications*, 181(9):1549–1556, 2010.

[4] Stephen G Brush. History of the lenz-ising model. *Reviews of modern physics*, 39(4):883, 1967.

[5] Jun Cai, William G Macready, and Aidan Roy. A practical heuristic for finding graph minors. *arXiv preprint arXiv:1406.2741*, 2014.

[6] Vicky Choi. Minor-embedding in adiabatic quantum computation: Ii. minor-universal graph design. *Quantum Information Processing*, 10(3):343–353, 2011.

[7] Caterina De Simone, Martin Diehl, Michael Jünger, Petra Mutzel, Gerhard Reinelt, and Giovanni Rinaldi. Exact ground states of ising spin glasses: New experimental results with a branch-and-cut algorithm. *Journal of Statistical Physics*, 80(1-2):487–496, 1995.

[8] Jack Edmonds. Maximum matching and a polyhedron with 0, l-vertices. *Journal of Research of the National Bureau of Standards, B. Mathematics and Mathematical Physics*, 69(1965):125–130, 1965.

[9] A Gilman, A Leist, and KA Hawick. 3d lattice monte carlo simulations on fpgas. In *Proceedings of the International Conference on Computer Design (CDES)*, page 1. The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp), 2013.

[10] MW Johnson, MHS Amin, S Gildert, T Lanting, F Hamze, N Dickson, R Harris, AJ Berkley, J Johansson, P Bunyk, et al. Quantum annealing with manufactured spins. *Nature*, 473(7346):194–198, 2011.

[11] Sera Kahruman, Elif Kolotoglu, Sergiy Butenko, and Illya V Hicks. On greedy construction heuristics for the max-cut problem. *International Journal of Computational Science and Engineering*, 3(3):211–218, 2007.

[12] Scott Kirkpatrick. Optimization by simulated annealing: Quantitative studies. *Journal of statistical physics*, 34(5-6):975–986, 1984.

[13] Tal Levy, Guy Cohen, and Eran Rabani. Simulating lattice spin models on graphics processing units. *Journal of chemical theory and computation*, 6(11):3293–3301, 2010.

[14] Y Lin, F Wang, X Zheng, H Gao, and L Zhang. Monte carlo simulation of the ising model on fpga. *Journal of Computational Physics*, 237:224–234, 2013.

[15] Andrew Lucas. Ising formulations of many np problems. *arXiv preprint arXiv:1302.5843*, 2013.

[16] Shoko Utsunomiya, Kenta Takata, and Yoshihisa Yamamoto. Mapping of ising models onto injection-locked laser systems. *Optics express*, 19(19):18091–18108, 2011.

[17] Martin Weigel. Simulating spin models on gpu. *Computer Physics Communications*, 182(9):1833–1836, 2011.

[18] M. Yamaoka, C. Yoshimura, M. Hayashi, T. Okuyama, H. Aoki, and H. Mizuno. A 20k-spin ising chip to solve combinatorial optimization problems with cmos annealing. *IEEE Journal of Solid-State Circuits*, 51(1):303–309, Jan 2016.

[19] Chihiro Yoshimura, Masanao Yamaoka, Hidetaka Aoki, and Hiroyuki Mizuno. Spatial computing architecture using randomness of memory cell stability under voltage control. In *2013 European Conference on Circuit Theory and Design (ECCTD)*, pages 1–4. IEEE, 2013.