PARALLELIZATION OF WOLFF ALGORITHM FOR LARGE-SCALE HIGH-DIMENSIONAL ISING MODEL

DIFEI ZHANG*

Abstract. The Ising model exhibits phase transitions and critical behavior that are crucial to understand complex systems. The Metropolis-Hastings algorithm and the Wolff algorithm are commonly used for sampling the Ising model. In this report, we provide a concise overview of these algorithms and implement a novel approach for parallelizing the Wolff algorithm. We leverage boundary partitioning and multi-threading techniques to distribute the workload among parallel threads, addressing the challenges of boundary conditions and high communication costs. Our parallelization scheme enhances the computational efficiency of the Wolff algorithm and enables the exploration of larger-scale higher-dimensional Ising models.

Key words. Julia, parallel computing, boundary partitioning, multi-threading, Monte Carlo, Wolff algorithm, Ising model

1. Introduction.

1.1. Ising Model. The Ising model is a fundamental statistical lattice model wherein each site represents a binary value, typically denoting spin up or spin down. Despite its conceptual simplicity, the Ising model finds wide-ranging applications in statistical physics, Bayesian inference, and numerous other scientific domains [5]. The efficient generation of the distribution of the large-scale high-dimensional Ising model plays a pivotal role in conducting precise simulations and facilitating comprehensive analyses.

1.2. Phase Transition and Critical Temperature. The Ising model exhibits intriguing phase transitions, characterized by abrupt changes in its properties as temperatures vary. As depicted in Figure 1, the average magnetization (the average spin) of the lattice experiences a significant decline as the temperature rises. At very low temperatures, all spins tend to align in the same direction, while at very high temperatures, the spin configuration becomes uniformly random and disordered. However, at a specific critical temperature in middle (denoted by the red dashed line), the system undergoes a phase transition, triggering the emergence of long-range correlations and collective phenomena, marking a coexistence of both global order and local disorder.

The comprehension of phase transitions and critical behavior assumes paramount importance in the study of complex systems and bears profound implications across diverse scientific disciplines. However, conventional sampling algorithms encounter significant challenges in the critical region, resulting in the known critical slowdown. Moreover, as the lattice size and dimension increase, the performance further deteriorates. Consequently, there is a compelling need to make the most efficient sampling algorithm parallelizable to address these limitations.

2. The Metropolis-Hastings Algorithm: 1-spin Flip.

2.1. Markov Chain and Stationary Distribution. The Metropolis-Hastings algorithm, a widely adopted Markov chain Monte Carlo (MCMC) method, is instrumental in sampling from complex distributions. Central to this algorithm is the concept of a Markov chain, which represents a sequence of random variables where the probability distribution of the next state is solely dependent on the current state. The key objective of the Metropolis-Hastings algorithm is to ensure that the Markov chain

*MIT EECS, Cambridge, MA (difeiz@mit.edu).

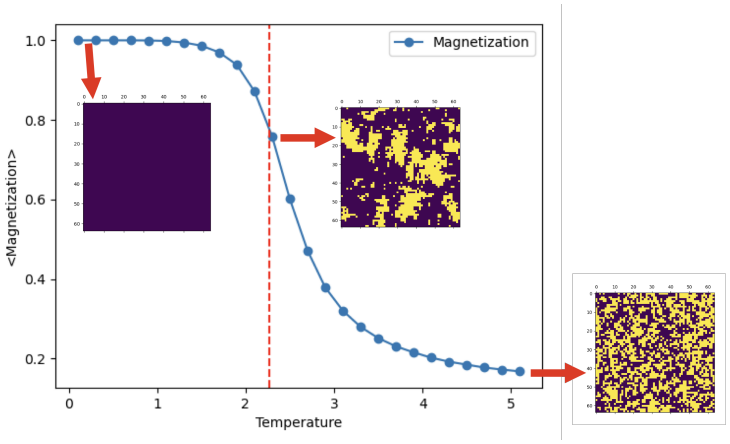


FIG. 1. *Phase transition demonstration for a* 64×64 *Ising lattice.*

converges to its stationary distribution, enabling correct sampling from the desired distribution.

Achieving convergence to the stationary distribution relies on two essential conditions: ergodicity and detailed balance. Ergodicity ensures that the Markov chain can explore the entire configuration space, allowing for the comprehensive sampling of possible states. Detailed balance, on the other hand, ensures that the transition probabilities between states satisfy the equilibrium condition, thereby guaranteeing that the stationary distribution is indeed reached.

The principle of detailed balance establishes a relationship between the transition probabilities from one configuration to another in a Markov chain [4, 1]. In the context of the Ising model, detailed balance relates the probability of transitioning from configuration C to a new configuration C', denoted as $W[C \to C']$, with the probability of transitioning in the reverse direction, $W[C' \to C]$. Additionally, it involves the probabilities $P[C]$ and $P[C']$ of observing the configurations C and C', respectively, within the ensemble of all possible configurations.

According to detailed balance, the ratio of these transition probabilities should be equal to the ratio of the probabilities of the configurations themselves, expressed as:

$$\frac{W[C \to C']}{W[C' \to C]} = \frac{P[C']}{P[C]}.$$

In the Ising model, the probabilities of configurations can be written in terms of the action, a quantity related to the energy of the system. This allows us to rewrite the detailed balance condition as:

$$\frac{W[C \to C']}{W[C' \to C]} = \frac{\exp(-E[C']/kT)}{\exp(-E[C]/kT)} = \exp(-\Delta E/kT),$$

where ΔE represents the energy change between configurations C and C', k is the Boltzmann constant and T is the temperature. This formulation demonstrates that the detailed balance condition relies on the exponential of the change in the energy, reflecting the energetic favorability of transitioning from one configuration to another.

2.2. Description of Metropolis-Hastings Sampling on Ising Model. The Metropolis-Hastings algorithm can be applied to sample from the Ising model by

proposing changes in the spin configurations and accepting or rejecting these changes based on a defined acceptance criterion. The iterative process (Figure 2) below allows for the systematic exploration of the configuration space and the generation of samples that follow the desired Boltzmann distribution.

step 1. Randomly selecting a single site from the lattice to attempt an update. This site represents an individual spin in the Ising model.

step 2. Calculating the energy change ΔE caused by flipping the spin at the selected site. The energy change ΔE is computed by considering the interactions between the neighboring spins and the external magnetic field, according to the Hamiltonian of the Ising model, $H(C) = -J \sum_{<i,j>} s_i s_j$.

step 3. If $\Delta E \leq 0$, indicating a decrease in the system's energy, the spin is flipped with a probability of 11. This change is accepted, as it leads to a lower energy state.

step 4. If $\Delta E > 0$, implying an increase in the system's energy, the spin is still potentially flipped, but the acceptance is determined by a probability criterion. The flip is accepted with a probability equal to $\exp(-\Delta E/kT)$.

step 5. The algorithm loops back to step 1, repeating the process for a specified number of iterations or until a convergence criterion is met.

Correctness. The proof for the algorithm correctness is straightforward. If the configuration update $C \to C'$ is in $\Delta E > 0$ direction, based on step 4, the probability $W[C \to C'] = \exp(-\Delta E/kT)$.Consequently,thereversedupdatemustbein. Consequently, the reversed update must be in $\Delta E < 0$ direction,fromstep3,theprobability direction, from step 3, the probability $W[C' \to C] = 1$. Therefore, it follows that

$$\frac{W[C \to C']}{W[C' \to C]} = \frac{\exp(-\Delta E/kT)}{1} = \exp(-\Delta E/kT).$$

We skip the proof for the other case for the symmetry and simplicity.

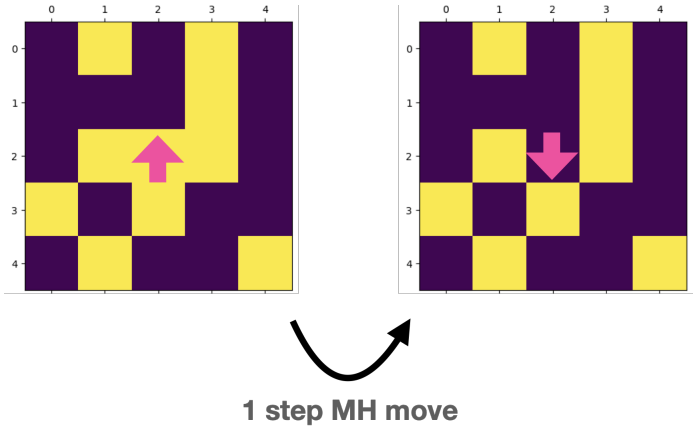


FIG. 2. *Metropolis Hasting Sampling demonstration for a* 5×5 *Ising lattice.*

3. The Wolff Algorithm: n-spin Flip.

3.1. Description of Wolff Cluster Sampling on Ising Model. The Wolff algorithm [6, 7] stands as a highly efficient and powerful method for sampling the Ising model, particularly when dealing with large-scale and high-dimensional systems. This algorithm introduces a unique approach by flipping clusters of spins instead of individual spins, resulting in a more effective exploration of the configuration space. By exploiting the inherent correlations among local clusters (depiced in the middle inset of Figure 1), the Wolff algorithm demonstrates faster convergence and significantly mitigates the issue of critical slowing down that commonly arises near phase transitions, distinguishing it from other sampling algorithms.

The key steps involved in the Wolff cluster sampling algorithm for the Ising model can be summarized as follows and depicted in Figure 3:

step 1. Randomly choose a seed spin from the lattice, serving as the initial spin of the new cluster. This seed spin is flipped to initiate the cluster formation.

step 2. Sequentially examine each neighboring spin of the seed spin. Specifically, identify those neighboring spins that point in the opposite direction as the flipped seed spin. For each of these anti-aligned spins, determine their inclusion in the growing cluster based on a probability threshold. The probability of adding an anti-aligned spin to the cluster is typically determined by the probability $1 - \exp(-2\psi_i\psi_j/kT)$, where $\psi_i\psi_j$ is the interaction between two linked spins. Here, ψ_i is the seed spin and ψ_j is the candidate spins.

step 3. For each newly added spin to the cluster in the previous step, perform a similar examination of its neighboring spins. Once again, identify the anti-aligned spins among these neighbors and determine their inclusion in the cluster using the same probability threshold as in step 2. This iterative process allows the cluster to grow by incorporating additional anti-aligned spins.

step 4. Repeat step 3 until there are no new spins added to the cluster. This termination condition ensures that the cluster has reached its maximal size and captures the connected region of anti-aligned spins.

Correctness. The Wolff algorithm also satisfies the condition of detailed balance. We denote the cluster by the inner $\mathcal{C}$ and the boundary $\partial\mathcal{C}$. Since the probability of flipping a cluster is 1, we only need to calculate the total probability of building a certain given cluster. In step 1, we randomly choose the seed spin, and therefore the probability P_{seed} is the same. Then the probability of building a given cluster is the product of the probabilities p_+ that each internal link $\ell_+ \in \mathcal{C}$ will be activated, times the product of the probabilities q_+ that each boundary link $\ell_+ \in \partial\mathcal{C}$ will not be activated, where $q_+ = 1 - p_+ = \exp(-2\psi_-\psi_+/kT)$. Therefore the complete probability for the update of the configuration is

$$W[C \to C'] = P_{seed} \left(\prod_{\ell_+ \in \mathcal{C}} p_+ \right) \left(\prod_{\ell_+ \in \partial\mathcal{C}} q_+ \right),$$

and the revered update probability is

$$W[C' \to C] = P_{seed} \left(\prod_{\ell_+ \in \mathcal{C}'} p_+ \right) \left(\prod_{\ell_+ \in \partial\mathcal{C}'} q_+ \right).$$

Noticing that for two configuratons, the inner part interaction is the same, and the only difference is the reversion of the boundary interaction, thus we can derive

$$\frac{W[C \to C']}{W[C' \to C]} = \frac{\prod_{\ell_+ \in \partial \mathcal{C}} \exp(-2\psi_- \psi_+ / kT)}{\prod_{\ell_+ \in \partial \mathcal{C}'} \exp(-2\psi_- \psi_+ / kT)} = \frac{\prod_{\ell \in \partial \mathcal{C}'} \exp(\psi_- \psi_+ / kT)}{\prod_{\ell \in \partial \mathcal{C}} \exp(\psi_- \psi_+ / kT)}.$$

Continuing this derivation, we establish the detail balance condition

$$\frac{W[C \to C']}{W[C' \to C]} = \frac{\exp(\sum_{\ell \in \partial \mathcal{C}'} \psi_- \psi_+ / kT)}{\exp(\sum_{\ell \in \partial \mathcal{C}} \psi_- \psi_+ / kT)} = \exp(-\Delta E / kT).$$

By repeatedly executing these steps, the Wolff algorithm effectively samples the configuration space of the Ising model, generating a series of spin configurations that approximate the desired distribution. The algorithm's ability to capture large clusters enhances the detection of phase transitions and critical phenomena, and it has been widely adopted in various fields of computational physics due to its superior performance in simulating Ising models. In the following sections, we will delve into the parallelization of the Wolff algorithm, employing boundary partitioning and multi-threading techniques, to further enhance its computational efficiency and enable the exploration of even larger-scale higher-dimensional Ising models.

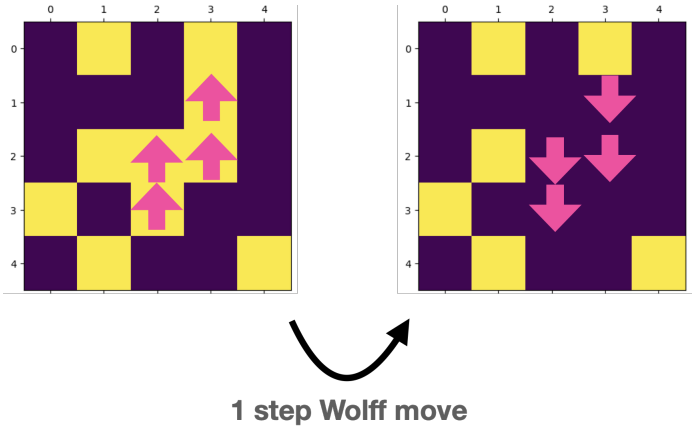


FIG. 3. *Wolff Sampling demonstration for a* 5×5 *Ising lattice.*

4. Parallelization of Wolff Algorithm.

4.1. Motivation. The Wolff algorithm has established itself as a highly efficient method for sampling high-dimensional and large-scale Ising models. However, parallelizing this algorithm presents notable challenges due to the inherent complexities of its implementation. Conventional parallelization approaches [2], such as lattice partitioning combined with message passing interface (MPI), encounter difficulties associated with boundary condition issues and the high communication costs incurred during the parallel execution. In light of these challenges, we implement a novel approach [3] to parallelize the Wolff algorithm by leveraging boundary partitioning and multi-threading techniques, which hold promise in overcoming the aforementioned limitations.

4.2. Boundary Partitioning in Step 3 of Serial Algorithm. The parallelization of the Wolff algorithm involves a carefully designed scheme utilizing boundary partitioning and multi-threading to distribute the workload efficiently among parallel threads. The following steps outline the process in detail (Figure 4):

step 3.1: Divide the List of the Growing Cluster Boundary. In this step, the list comprising the growing cluster boundary is divided among the parallel threads. This division ensures that each thread is responsible for handling a specific fraction of the cluster boundary. By assigning a portion of the boundary to each thread, the workload is evenly distributed, maximizing the utilization of computational resources.

Furthermore, to facilitate seamless data access and synchronization between the threads, the coordinate array and array of spin variables are stored in shared memory. This shared memory space enables efficient communication and coordination among the parallel threads during the execution of the parallelized algorithm.

step 3.2: Parallel Searching in Each Direction. For effective parallelization, each thread independently performs Step 3 of the Wolff algorithm in parallel for a specific searching direction. In a three-dimensional lattice, this typically involves considering six directions. Although different directions searching are executed in serial, given a certain direction, the boundary is growing in parallel, efficiently exploring the configuration space of the Ising model. Within each thread, the assigned fraction of the cluster boundary, as determined in Step 1 of the parallelization scheme, is processed. Additionally, each thread forms its new boundary consisting of newly added spins.

To ensure proper synchronization and data integrity, the newly added spins, which are shared variables among the parallel threads, play a vital role. Each thread generates its own new boundary by examining neighboring spins and determining the ones pointing in the opposite direction. These spins are then added to the thread's new boundary. This process allows each thread to independently identify and gather relevant spins for further analysis and cluster growth.

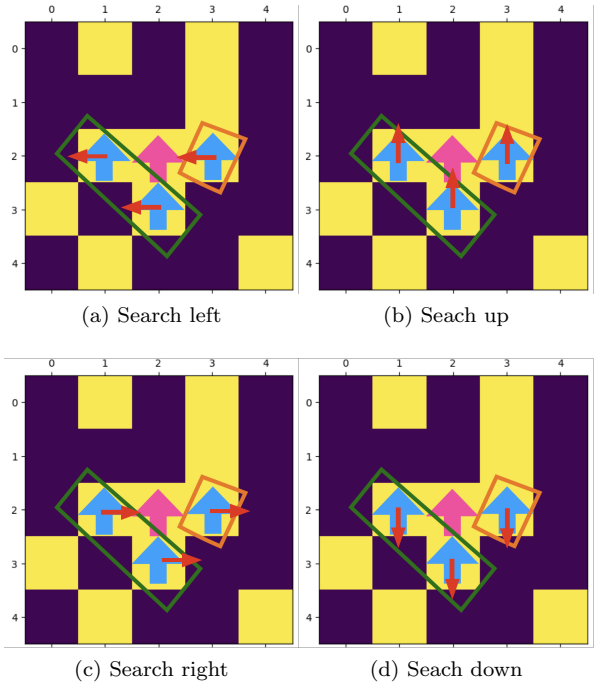
step 3.3: Formation of the Common New Cluster Boundary. After the parallel execution of Step 3 in multiple directions, it is necessary to form a common new cluster boundary that incorporates the newly added spins from all threads. This step involves merging the individual thread's new boundaries, combining the pertinent information from each thread's local computation.

The merging process ensures that the common new cluster boundary reflects the collective contributions of all parallel threads. By consolidating the newly added spins, the common boundary represents an updated and comprehensive set of spins for further iterations of the parallelized algorithm.

Upon completing the formation of the common new cluster boundary, the scheme returns to Step 1, initiating the next iteration of the parallelized algorithm. This iterative process continues until convergence is achieved or a predetermined stopping criterion is met.

Correctness. The correctness of the parallelized Wolff sampling can be demonstrated by showing that multiple threads accessing and modifying the shared memory do not lead to conflicts. Given a certain searching direction, since no new spin can be accessed twice, we do not need to concern such conflict. For example, in a two-dimensional lattice A, the lattice site $A[i, j]$ can only be visited and modified from $A[i-1, j], A[i+1, j], A[i, j-1], A[i, j+1]$ in four different directions rather than the same direction.

When transitioning from one search direction to the next, three possible cases arise for the spins to be searched: those never seen before, those visited but not added, and those visited and added. In the first case, a normal search is performed, and the spin is added to the shared memory if necessary. In the second case, the spin has been



(a) Search left (b) Seach up

(c) Search right (d) Seach down

FIG. 4. *Parallelization design of the Wolff algorithm demonstration for a* 5×5 *Ising lattice. Search each direction in serial. In each direction, the current boundary (blue spins) are partitioned and fed into different threads (orange and green), allowing parallel searching.*

visited in the previous direction search but was not added to the boundary or shared memory. In this case, the spin is given another chance to be activated according to the determined probability $1 - \exp(-2\psi_i\psi_j/kT)$. Adding it to the shared memory does not lead to conflicts.

In the third case, where the spin has already been added to the shared memory, each thread has access to the shared memory, and the direction searches are executed sequentially. Therefore, when encountering a spin that has been marked in the shared memory, the thread knows to skip it without causing any conflicts. Overall, by allowing each thread access to the shared memory, it is ensured that conflicts do not occur.

Thus, based on no issue by assigning shared memory, plus the similar proof to that of that of the serial version, the parallelized Wolff sampling maintains correctness.

5. Results and Analysis.

5.1. Correctness of the Parallel Wolff Algorithm. We first demonstrate the correctness of our parallel version of Wolff sampling. For an $8times8$ Ising lattice, we use Metropolis Hasting sampling, simulating 10000 times at each temperature, as a benchmark. Then for our parallel Wolff sampling, we set JULIA_NUM_THREADS = 1 as a serial version, meaning no boundary partitioning can happen. For each sampling method, we calculate the average energy $\langle E(T)\rangle = \langle -J\sum_{<i,j>} s_i s_j\rangle$. As is shown in Figure 5, the two overlapped curves demonstrate the stationary distributions are the same at least for the first moment, serving as a part proof for the correctness of the parallel Wolff sampling implementation.

We provide an analysis of the correctness of our parallelized Wolff sampling algorithm. To validate the parallel implementation, we conduct a benchmark comparison with the Metropolis-Hasting sampling method on an 8×8 Ising lattice. The Metropolis-Hasting sampling is performed with $10,000$ simulations at each temperature as a reference.

In our parallel Wolff sampling, we initially set JULIA_NUM_THREADS $= 1$, simulating the algorithm in a serial manner without any boundary partitioning. For both the Metropolis-Hasting and parallel Wolff sampling methods, we calculate the average energy $\langle E(T) \rangle = \langle -J \sum_{<i,j>} s_i s_j \rangle$.

The results, as illustrated in Figure 5, show that the two energy curves overlap, indicating that the stationary distributions are consistent at least for the first moment. This serves as preliminary evidence of the correctness of our parallel implementation of the Wolff sampling algorithm.

The comparison between the Metropolis-Hasting and parallel Wolff sampling methods provides insight into the fidelity and accuracy of the parallel implementation, supporting its correctness in capturing the essential features of the Ising model's energy distribution.

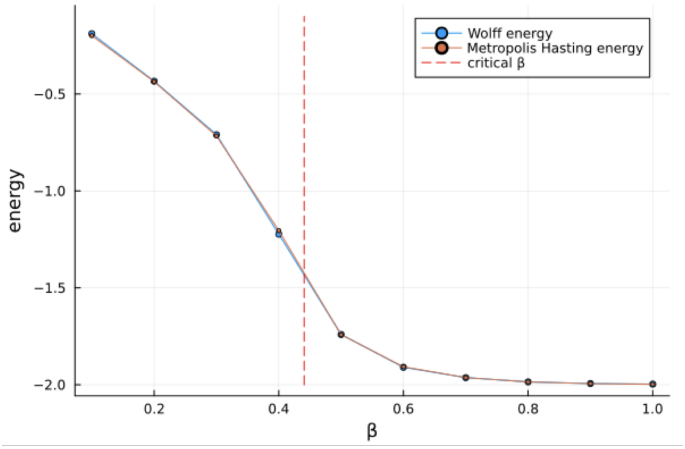


FIG. 5. *Correctness of parallel Wolff Sampling implementation on an* 8×8 *Ising lattice.*

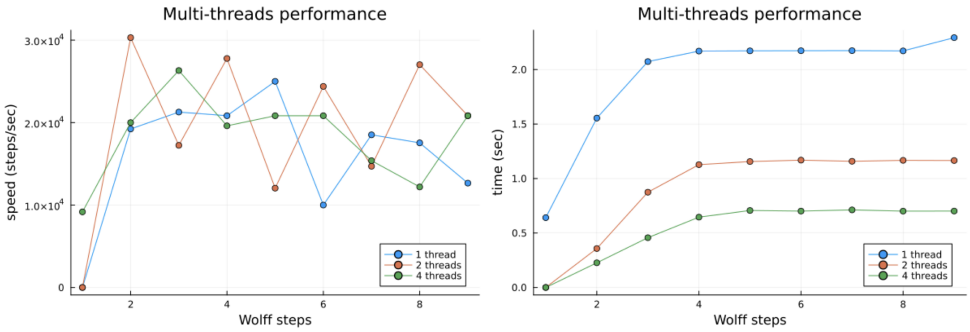
5.2. Comparison of Serial and Parallel Wolff Algorithm. In order to assess the speedup achieved by our parallel Wolff sampling implementation, we conducted experiments on lattices of size $262,144$. Specifically, we performed the experiments on two different systems: a two-dimensional lattice with dimensions 512×512 and a three-dimensional lattice with dimensions $64 \times 64 \times 64$. The parallelization was carried out by varying the number of threads, setting JULIA_NUM_THREADS $= 1$, 2, and 4 for the respective systems.

The performance evaluation focused on measuring the time consumption of each Wolff sampling step and observing the resulting performance plateau. Figure 6 illustrates the recorded time consumption (or speed) for each step. In the case of the two-dimensional lattice (Figure 6a), the parallel implementation did not yield any noticeable acceleration for such a large Ising lattice.

However, in the case of the three-dimensional lattice (Figure 6b), the speedup achieved by the parallel implementation was significant and aligned with our expec-

tations. Doubling the number of threads resulted in a twofold increase in the speed of each Wolff sampling step.

The experimental results highlight the dependence of the speedup on the lattice dimensionality and the associated computational complexity. The lack of acceleration observed in the two-dimensional case suggests that the parallelization overhead may not be justified for large-scale 2D Ising models. Conversely, the substantial speedup observed in the three-dimensional case demonstrates the effectiveness of the parallel Wolff algorithm for high-dimensional systems.



(a) 2D 512×512 Ising Lattice (b) 3D $64 \times 64 \times 64$ Ising Lattice

FIG. 6. *Performance comparison of serial and parallel Wolff sampling.*

5.3. Dimension Analysis: Impact of Boundary Partitioning. In order to shed light on the reasons behind the observed acceleration of the parallel Wolff sampling algorithm in higher dimensions, we conducted a detailed analysis and further experimentation. The key insight lies in the significance of efficient boundary partitioning, which relies on having a sufficiently large boundary length for parallelization.

Intuitively, one might expect the boundary length to increase as the cluster grows, irrespective of the dimensionality. However, it is important to note that the boundary subjected to partitioning in our algorithm is not the conventional boundary of the cluster, but rather the boundary formed by the newly added spins. In other words, even if the cluster size is 100×100 resulting in a normal boundary length of 400, the actual boundary length in our algorithm would be only 2 if there are merely 2 newly added spins. This distinction is crucial in understanding the behavior of the parallelization.

As illustrated in Figure 7, we can observe that despite having the same lattice size, the boundary length of a two-dimensional 512×512 system in each Wolff sampling step is consistently less than 10. Consequently, there is limited potential for multi-threading acceleration in this scenario. On the contrary, as the dimension increases to 3 and 4, the boundary length becomes significantly larger even with the same lattice size. This discrepancy perfectly explains why the parallel Wolff sampling method is better suited for large-scale high-dimensional Ising models.

The findings from this dimension analysis highlight the crucial role of boundary length in determining the efficacy of the parallelization strategy. It reinforces the observation that the parallel Wolff algorithm demonstrates optimal performance when applied to high-dimensional Ising models with large-scale lattice sizes. This insight opens up avenues for further exploration of boundary partitioning techniques and their impact on different system dimensions, enabling the development of more efficient

parallel sampling algorithms for complex lattice models.

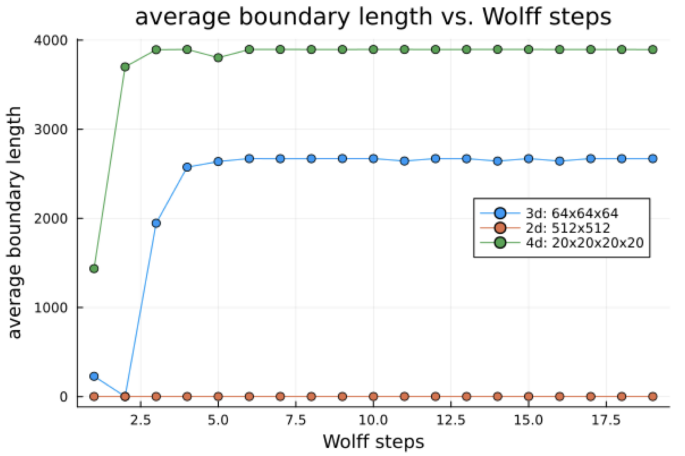


FIG. 7. *Dimension analysis for the parallel Wolff sampling on the same lattice size.*

6. Conclusion. In this report, we have explored the sampling of the Ising model using the Metropolis-Hastings algorithm and the Wolff algorithm. We have discussed the principles and implementation details of these algorithms, highlighting their correctness and their ability to generate samples following the desired Boltzmann distribution. The Metropolis-Hastings algorithm updates individual spins, while the Wolff algorithm flips clusters of spins, offering advantages in exploring the configuration space efficiently and mitigating critical slowing down near phase transitions.

Furthermore, we have implemented a novel approach to parallelize the Wolff algorithm by utilizing boundary partitioning and multi-threading techniques. This parallelization scheme allows for the efficient distribution of workload among parallel threads, maximizing computational resources' utilization. We have outlined the steps involved in the parallelization process, emphasizing the division of the growing cluster boundary and the parallel searching in each direction. In conclusion, the parallelization of the Wolff algorithm enhances its computational efficiency, enabling the exploration of larger-scale and higher-dimensional Ising models.

Further research can focus on optimizing the boundary partitioning strategy employed in the parallelized Wolff algorithm. Exploring different partitioning schemes, load balancing techniques, and communication strategies can enhance the algorithm's performance and scalability. Additionally, investigating adaptive partitioning methods that dynamically adjust the partition boundaries based on the evolving cluster structure may further improve the parallel efficiency. On the other hand, machine learning techniques, such as neural networks, have shown promise in enhancing Monte Carlo simulations. Integrating parallelized Wolff algorithm with machine learning approaches, such as generative models or reinforcement learning, can provide novel ways to explore complex phase spaces, optimize sampling strategies, and accelerate convergence, leading to more efficient simulations and deeper insights into statistical physics phenomena.

7. Supplemental Material. The Julia implementation of the parallel Wolff sampling can be found in the link https://github.com/difeizhang/ParallelWolff.git.

REFERENCES

[1] http://latt.if.usp.br/technical-pages/twawesab/Text.html/node1.html.

[2] D. Hassani and S. Rafibakhsh, *Parallelization and implementation of multi-spin monte carlo simulation of 2d square ising model using mpi and c++*, Journal of Theoretical and Applied Physics, 12 (2018), pp. 199–208, https://doi.org/10.1007/s40094-018-0301-4, https://doi.org/10.1007/s40094-018-0301-4.

[3] J. Kaupužs, J. Rimšāns, and R. V. N. Melnik, *Parallelization of the wolff single-cluster algorithm*, Phys. Rev. E, 81 (2010), p. 026701, https://doi.org/10.1103/PhysRevE.81.026701, https://link.aps.org/doi/10.1103/PhysRevE.81.026701.

[4] M. E. Newman and G. T. Barkema, *Monte Carlo methods in statistical physics*, Clarendon Press, 1999.

[5] S. P. Singh, *The ising model: Brief introduction and its application*, in Metastable, Spintronics Materials and Mechanics of Deformable Bodies, S. Sivasankaran, P. K. Nayak, and E. Gnay, eds., IntechOpen, Rijeka, 2020, ch. 8, https://doi.org/10.5772/intechopen.90875, https://doi.org/10.5772/intechopen.90875.

[6] R. H. Swendsen and J.-S. Wang, *Nonuniversal critical dynamics in monte carlo simulations*, Phys. Rev. Lett., 58 (1987), pp. 86–88, https://doi.org/10.1103/PhysRevLett.58.86, https://link.aps.org/doi/10.1103/PhysRevLett.58.86.

[7] U. Wolff, *Collective monte carlo updating for spin systems*, Phys. Rev. Lett., 62 (1989), pp. 361–364, https://doi.org/10.1103/PhysRevLett.62.361, https://link.aps.org/doi/10.1103/PhysRevLett.62.361.