Parallelization of Wolff Algorithm for Ising Model

Difei Zhang @MIT EECS
difeiz@mit.edu

May 1, 2023

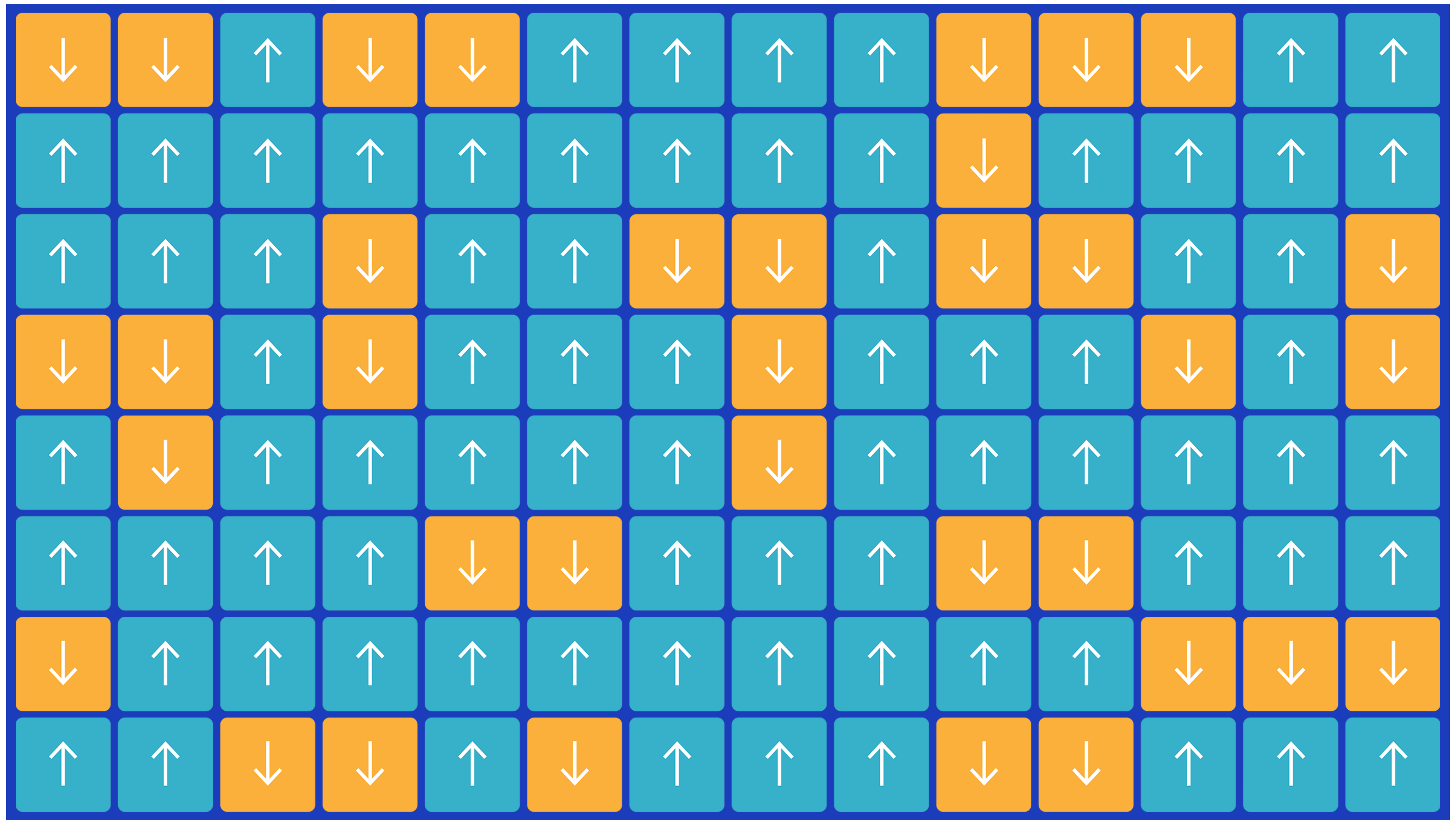
Contents

- Ising Model
- Metropolis Hasting Sampling
- Wolff Algorithm
- Parallelization of Wolff Algorithm

Ising Model

Introduction

- A statistical lattice model, each site is a binary value.
- Simple model, but wide application in statistical physics, inference, etc.
- Therefore, to efficiently generate the distribution of n-dim Ising model is important.

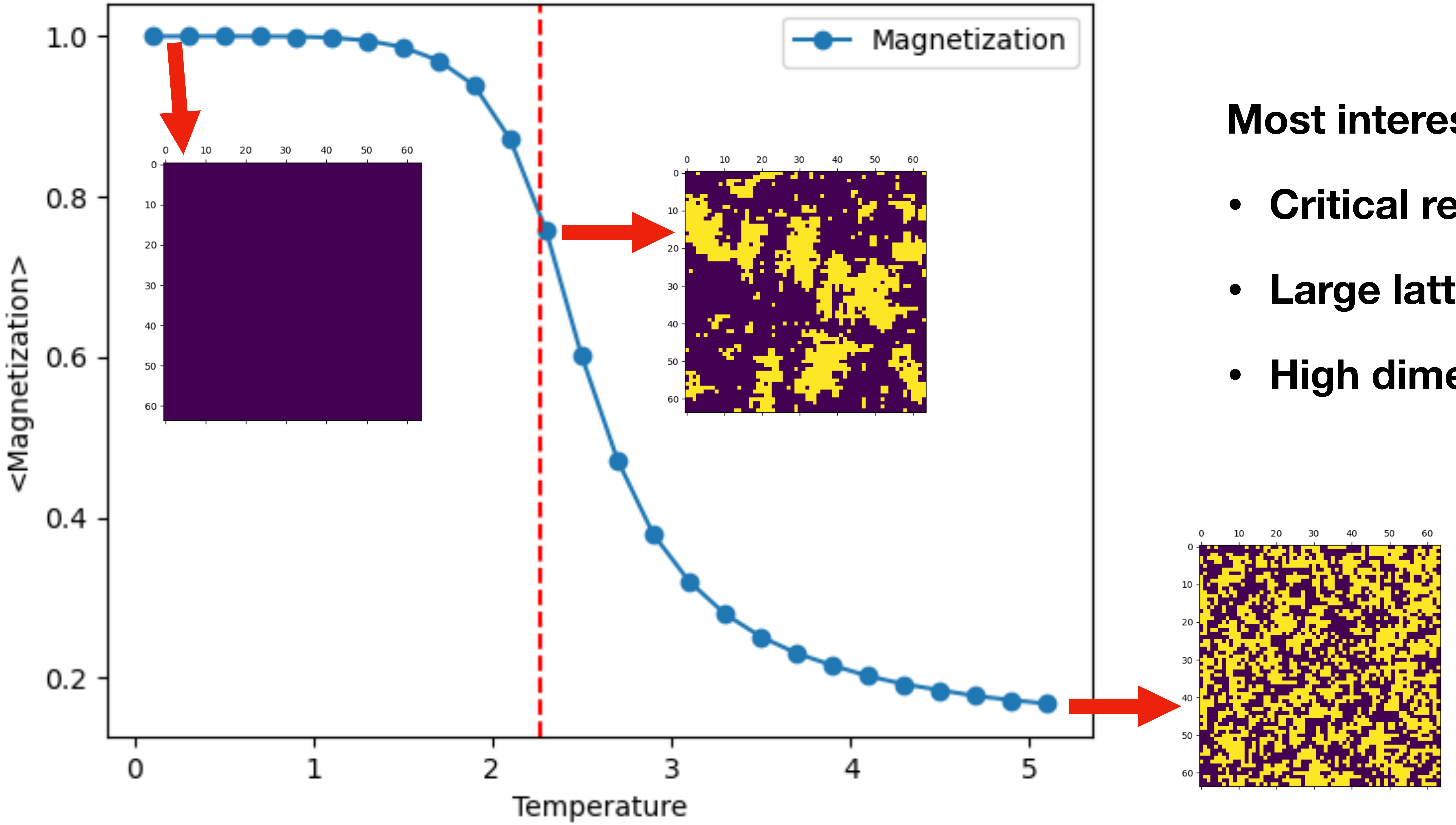


$$H(S) = -J \sum_{<i,j>} s_i s_j$$

$$P(S|T) = \frac{\exp(-H(S)/kT)}{Z}$$

https://www.quantamagazine.org/the-cartoon-picture-of-magnets-that-has-transformed-science-20200624/

Phase Transition & Critical Temperature



Most interesting but difficult:

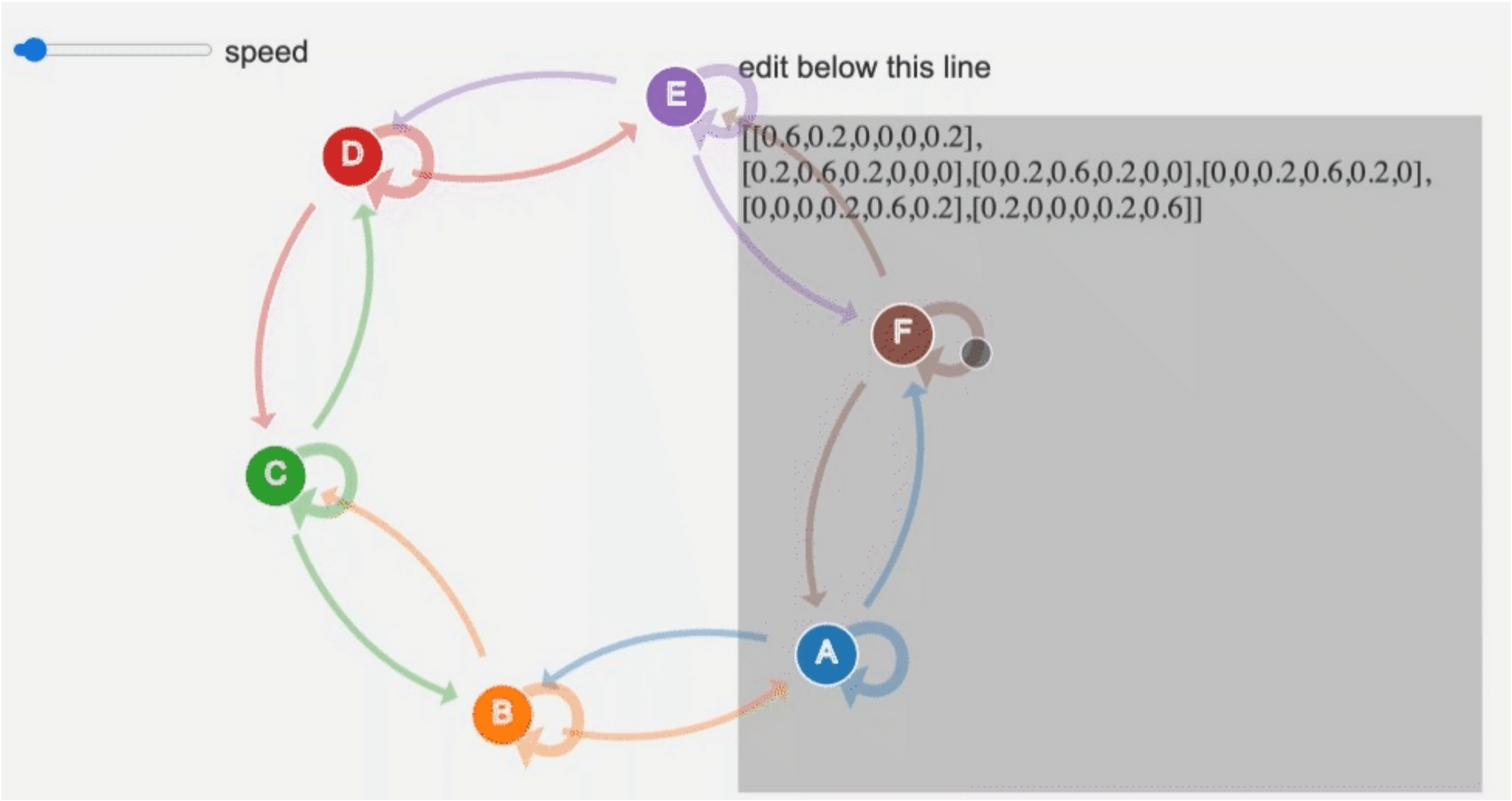
- **Critical region**
- **Large lattice length**
- **High dimension lattice**

Metropolis Hasting Sampling

Markov chain & Stationary Distribution

ergodicity + detailed balance

⇒ Markov Chain converges to stationary distribution



https://setosa.io/ev/markov-chains/

Metropolis Hasting Algorithm: 1-spin flip

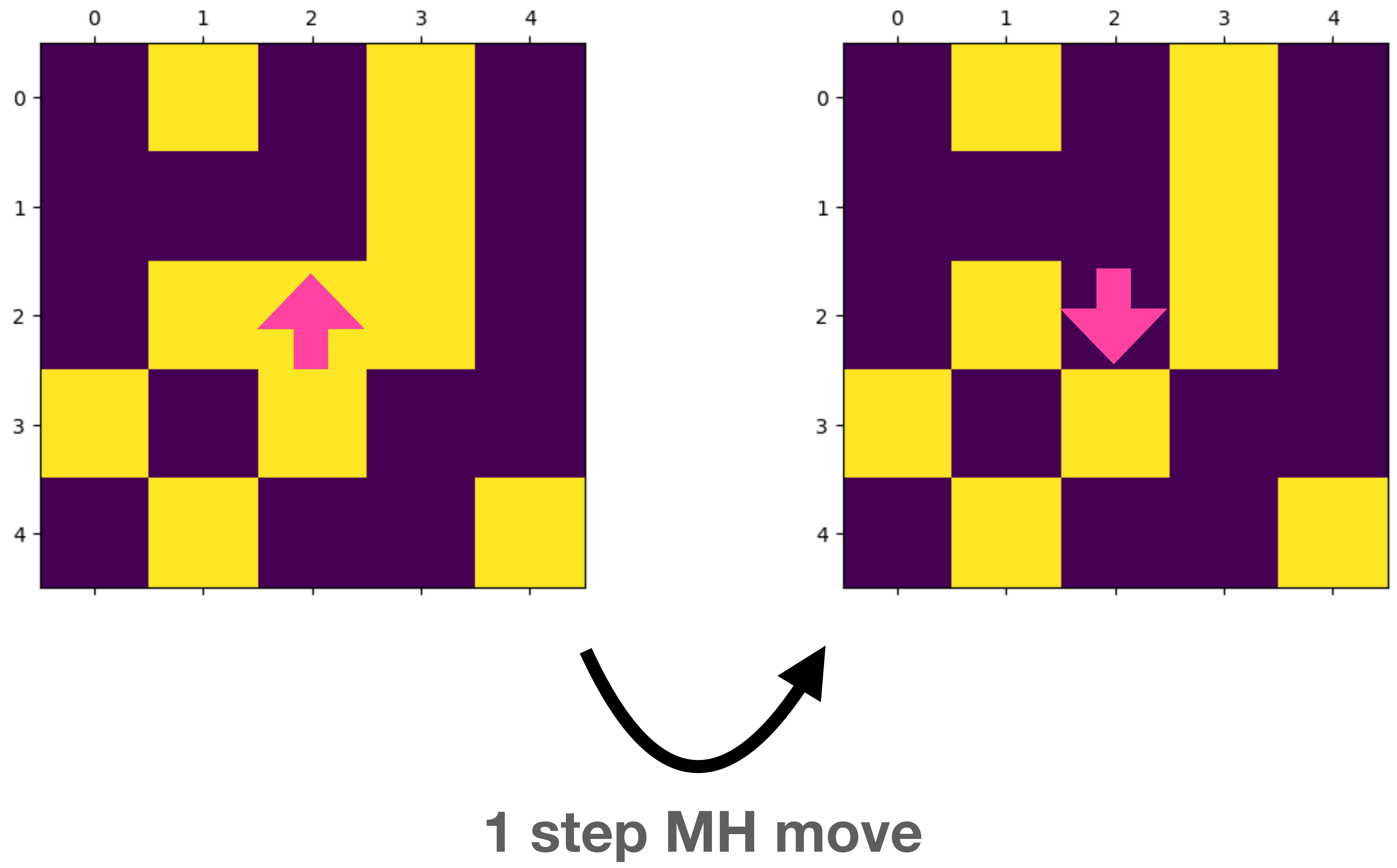
1. Randomly choose a single site of the lattice for trying an update.

2. Calculate the energy change ΔE caused by flipping the field at that site.

3. If $\Delta E \leq 0$, then flip the field with probability equal to 1.

4. If $\Delta E > 0$, then flip the field with probability $\exp(-\Delta E)$.

5. Loop back to step 1 and repeat the procedure.



Wolff Algorithm

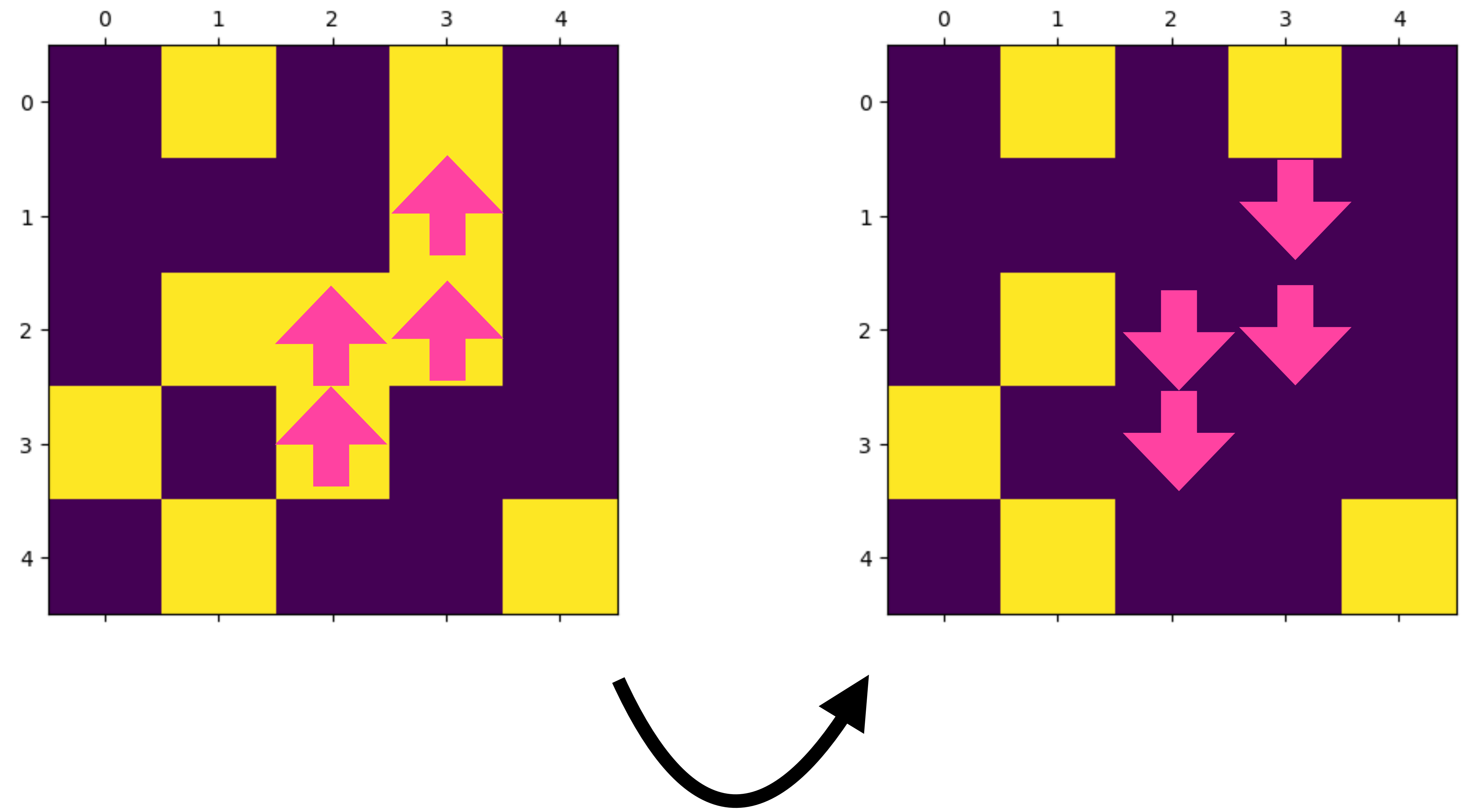
Wolff Algorithm: n-spin (cluster) flip

1. Randomly choose a seed spin of the new cluster and flip it. (Cluster seed)

2. Look in turn at each of the neighbors of the seed spin and find the ones pointing in the opposite direction as the flipped seed spin. Each of them is added to the cluster with probability $1 - \exp(-2\beta)$.

3. **For newly added spins in the last step, examine each of neighbors to find the ones pointing in the opposite direction, adding each of them to the cluster with probability $1 - \exp(-2\beta)$. (Growing cluster)**

4. Repeat step 3 until there are no new spins added to the cluster.



1 step Wolff move

Parallelization of Wolff Algorithm

Motivation

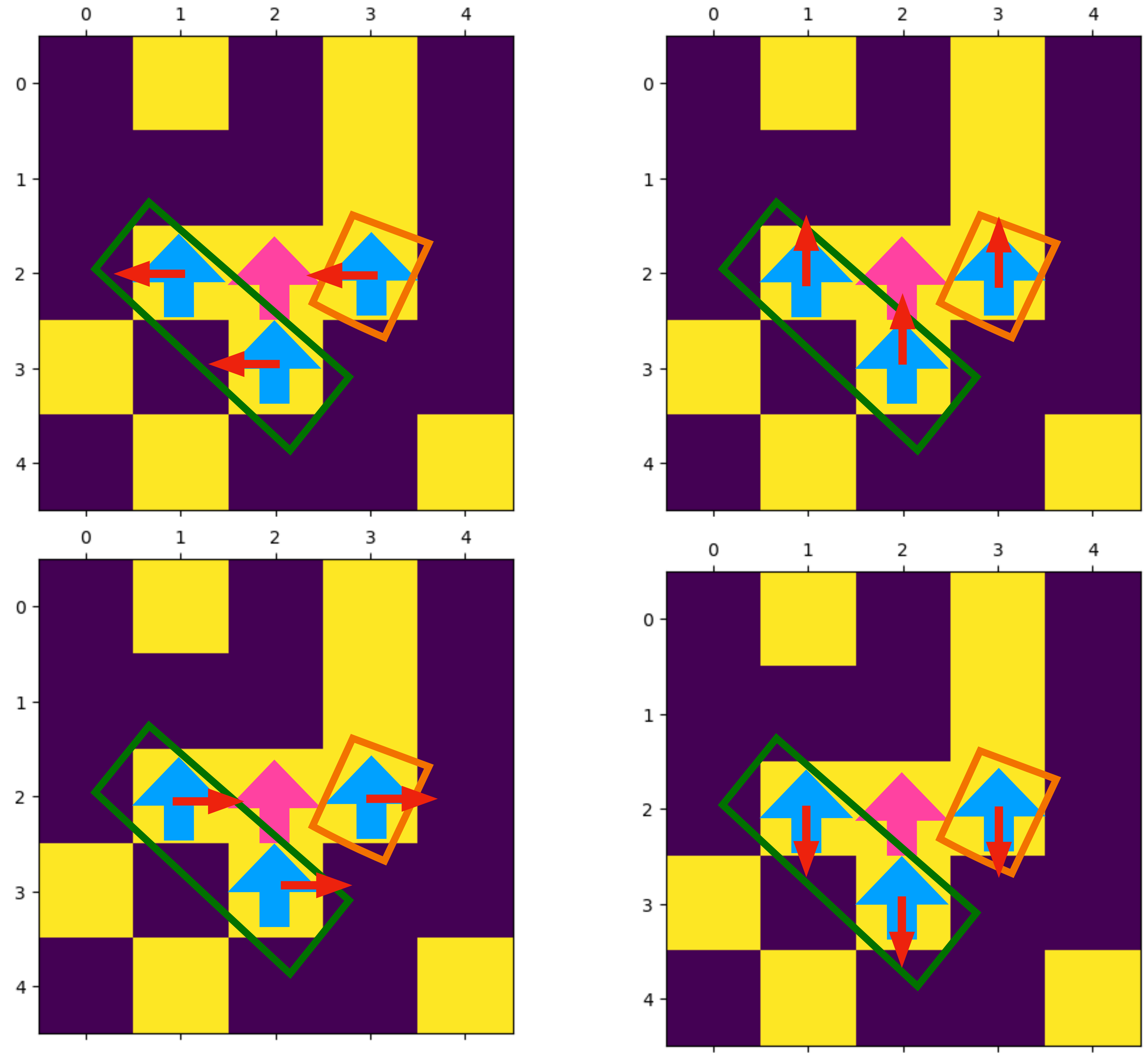
- Wolff Algorithm is the most efficient sampling method for high-dimension, large-scale Ising lattice, but also most difficult to parallelize.
- Conventional parallelization is done by **lattice partition** and **MPI**, but there exist boundary issue and communication cost.
- Here, we parallelize it by **boundary partition** and **multi-threading**.

Parallelization: boundary partition for step 3

1. Divide the list of the growing cluster boundary between the threads. Coordinate array and the array of spin variables are stored in the shared memory.

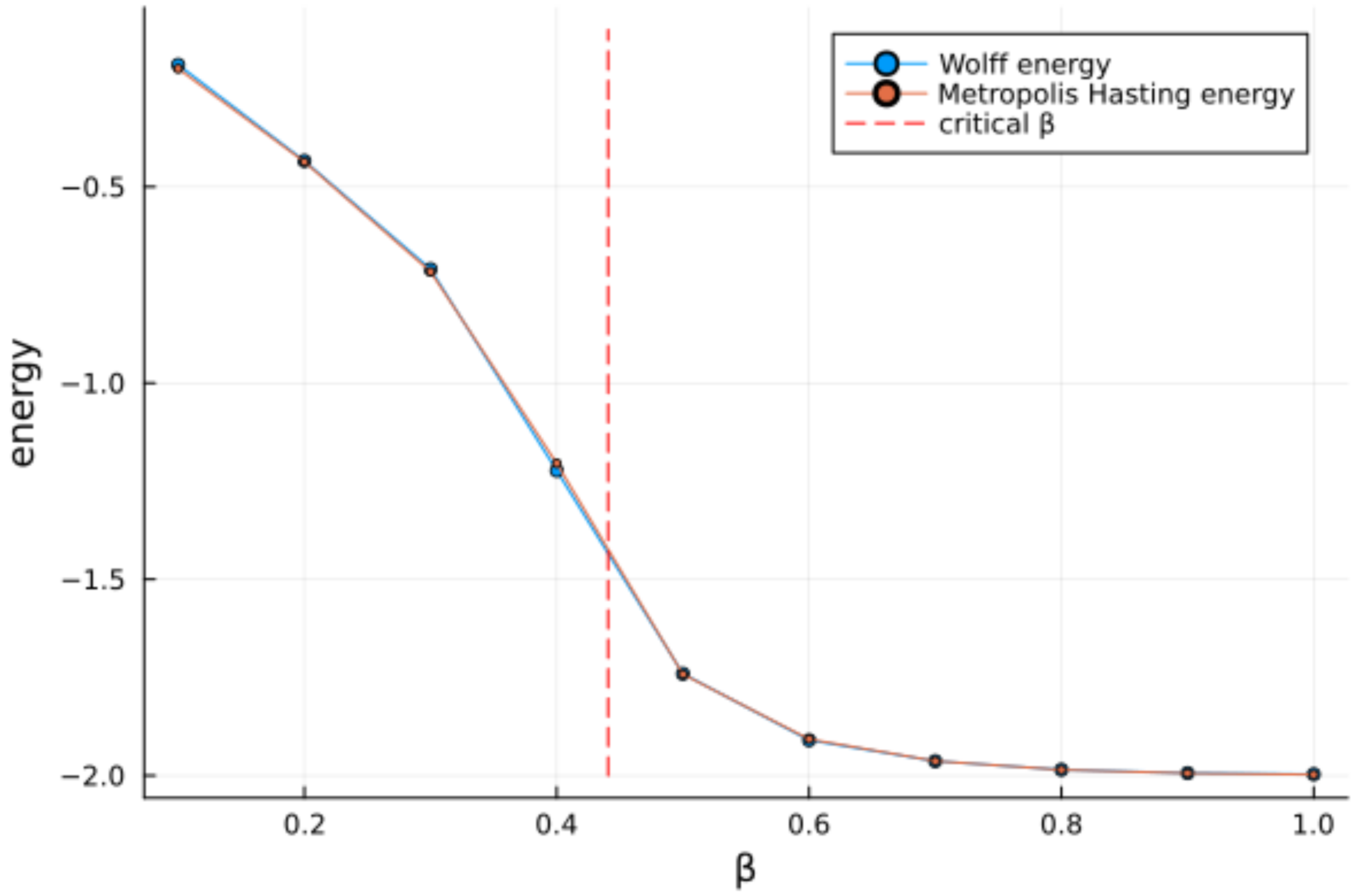
2. For each searching direction (e.g. 6 directions in 3D lattice), perform step 3 of the Wolff algorithm in parallel. Each thread treats certain fraction of the cluster boundary, assigned in step 1 of this scheme. Besides, each thread forms its new boundary of newly added spins, which are shared variables.

3. Form the common new cluster boundary with newly added spins, return step 1 in this scheme.



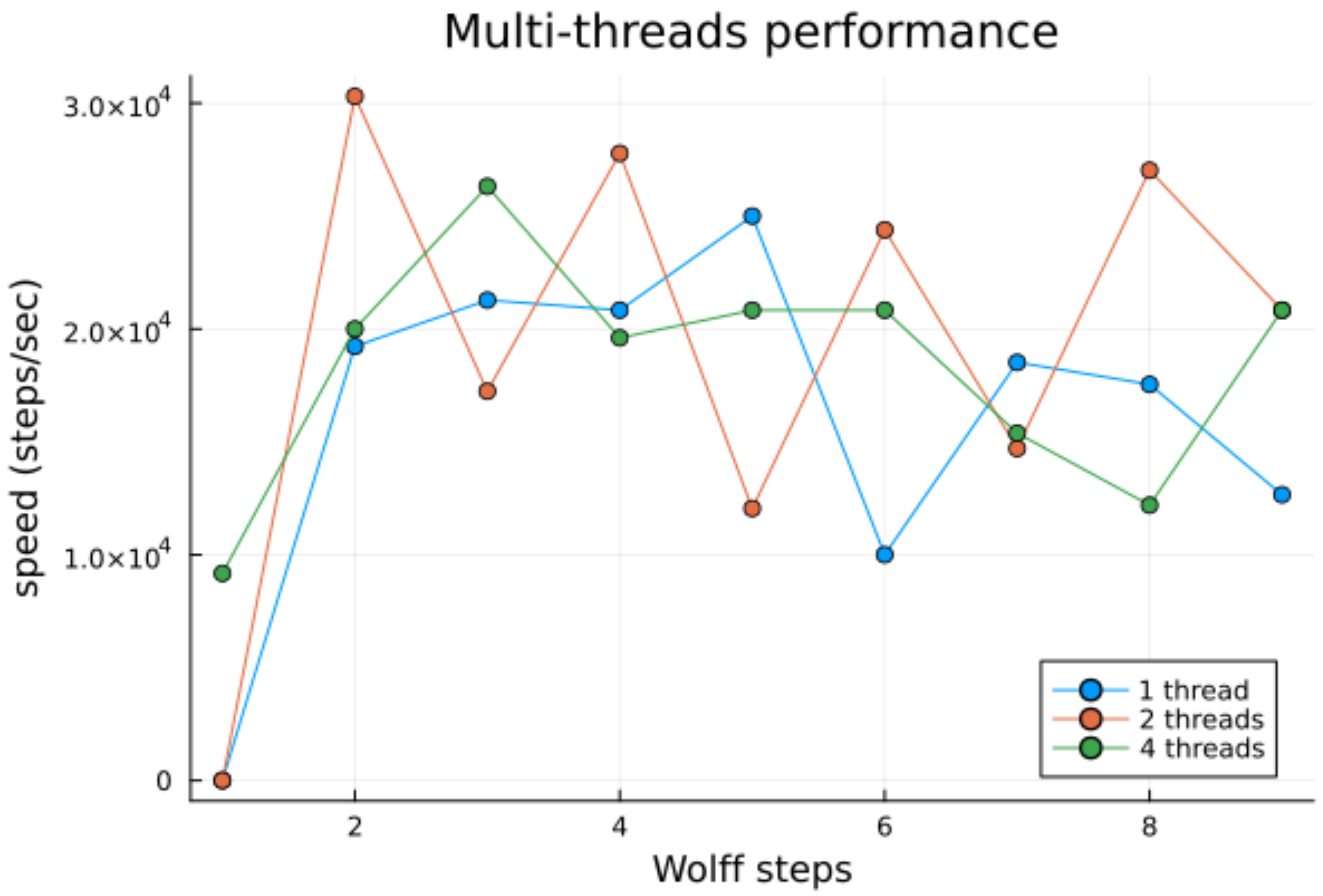
4 directions Wolff search: Search direction: red; Thread 1: Green; Thread 2: Orange.

Performance Analysis: Correctness

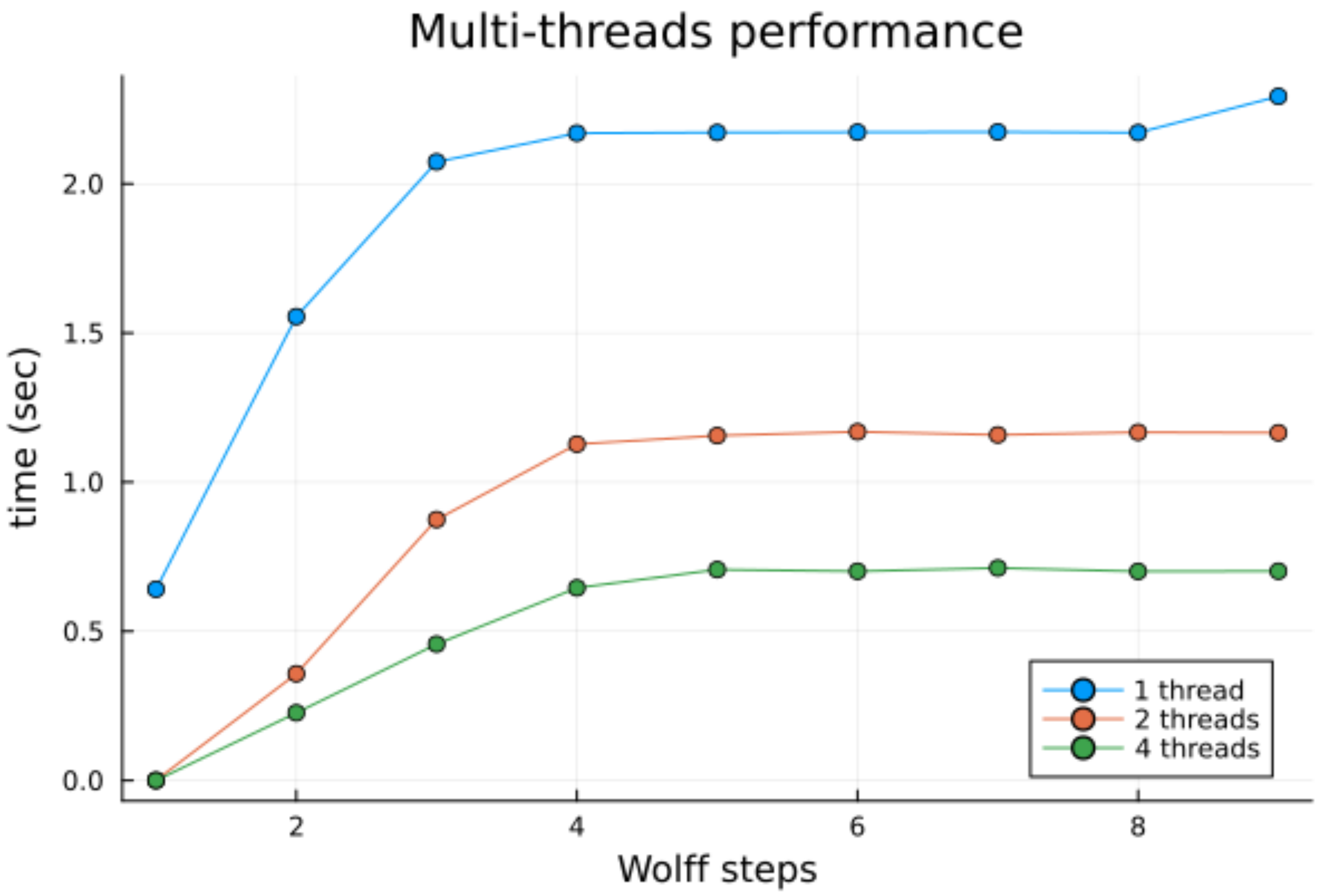


2-dim 8 x 8 Ising Lattice

Performance Analysis: operation time

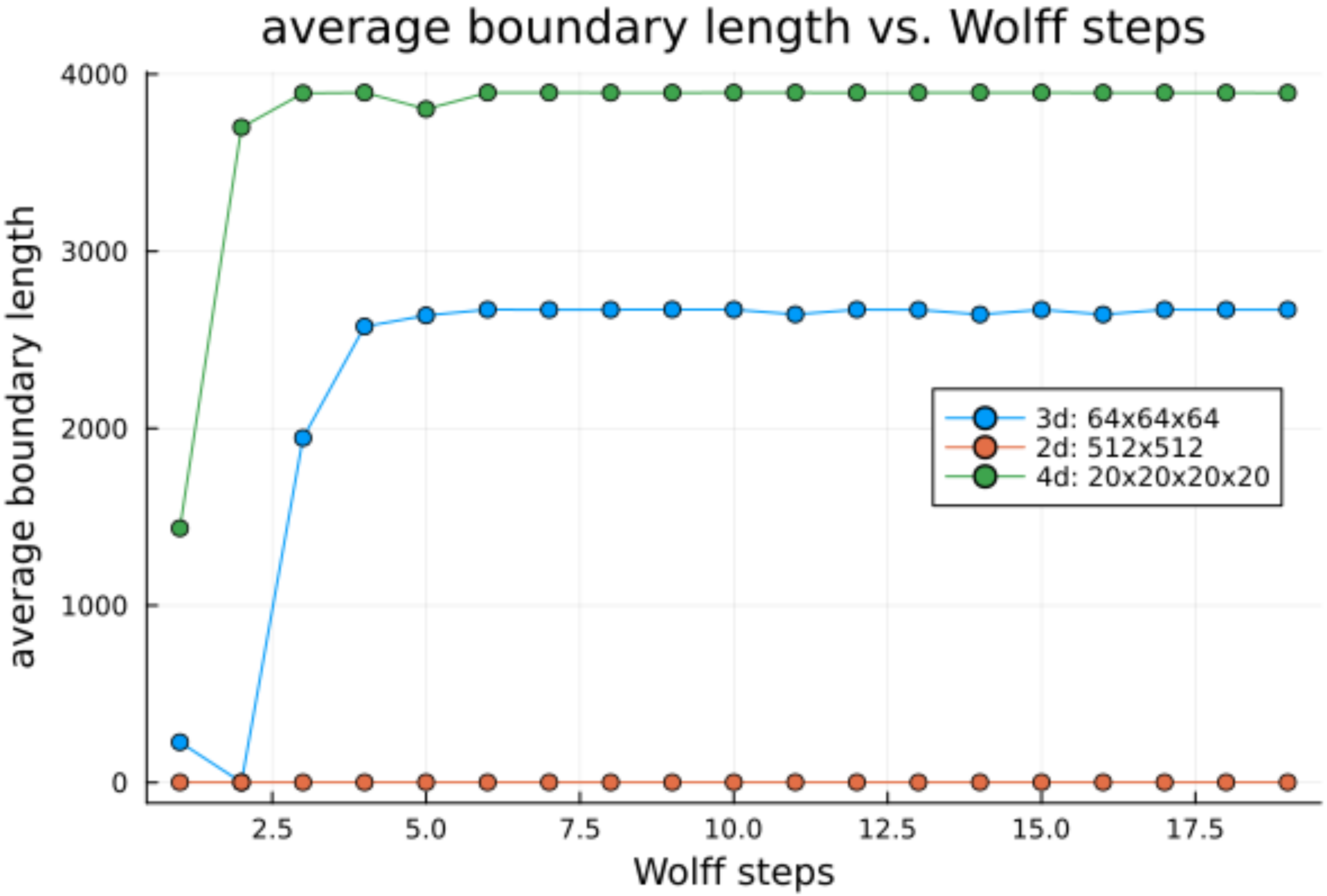


2-dim 512 x 512 Ising Lattice



3-dim 64 x 64 x 64 Ising Lattice

Performance Analysis: # dimensions



Future Work

Future Work

- Optimal boundary length C_{min} to determine when to do parallel computing
- Distribution in a temperature range $[T_{min}, T_{max}]$, not just single point $T \approx T_{crit}$
- Compute physics quantity (energy, magnetization, capacity, etc.) to analyze the quality of sampling

Reference

- Kaupužs, Jevgenijs, Jānis Rimšāns, and Roderick VN Melnik. "Parallelization of the Wolff single-cluster algorithm." Physical Review E 81.2 (2010): 026701.
- Newman, Mark EJ, and Gerard T. Barkema. Monte Carlo methods in statistical physics. Clarendon Press, 1999.

Thanks!