# Unbound Parallelization on Heisenberg Model

Arkavo Hait
*Department of Electronic Systems Engineering*
*Indian Institute of Science (IISc) Bangalore*
Bengaluru 560012, India
arkavohait@iisc.ac.in

Santanu Mahapatra
*Department of Electronic Systems Engineering*
*Indian Institute of Science (IISc) Bangalore*
Bengaluru 560012, India
santanu@iisc.ac.in

***Abstract*—Traditional Markov Chain Metropolis Monte Carlo (MCMC) simulations face severe scalability bottlenecks when applied to complex, large-scale magnetic systems. To overcome this, we introduce an "unbound" parallelization algorithm for the Heisenberg model, explicitly designed for modern GPU architectures using CUDA. By introducing a tunable Parallelization Ratio ($P$), our method decouples state evaluations, allowing the simulation to bypass the critical slowing down inherent to serial MCMC. This approach not only delivers massive computational acceleration but also enables the system to escape deep local energy minima that trap standard algorithms. We demonstrate the efficacy of this method by simulating micron-scale megastructures and complex spin textures, such as skyrmions, paving the way for rapid, fully in-silico material discovery.**

***Index Terms*—Monte Carlo, GPU, Python, CUDA, Heisenberg Model**

## I. Introduction

This work extends the analysis and capabilities of CUDA-METRO [1], a bespoke software framework designed to solve the Heisenberg model [2] utilizing Markov Chain Metropolis Monte Carlo (MCMC) methods [3] on 2D lattices. The motivation for this development stems from the scarcity of MCMC software optimized for modern Graphical Processing Units (GPUs). Existing solutions are often narrowly tailored to Ising models or restricted to small-scale square lattice configurations. Our code serves as a generalized template for evaluating MCMC-based systems on the NVIDIA-CUDA [4] backend.

The core innovation of this approach lies in the decomposition of the global state into independent paths for evaluation. This compartmentalization facilitates massive parallelization, making the algorithm highly amenable to GPU implementation. We designate this algorithm as "unbound" to denote that the theoretical acceleration is not constrained by serial dependencies, but rather by available hardware resources.

It is important to note that this parallelization strategy deviates from the strict statistics of detailed balance inherent to sequential MCMC. As the algorithm moves further from the serial approach, this deviation must be carefully calibrated to ensure physical validity. In our previous work [3], we validated, benchmarked, and tested this algorithm against various real-world scenarios. Here, we present advanced results and analyses derived from large-scale simulations, focusing on the trade-offs between parallel throughput and ergodic sampling.

## II. Algorithm

### A. Implementation

The algorithm operates by proposing changes to a large subset of the lattice simultaneously and evaluating the partition function for these changes independently. The system state is represented as a vector containing atomic spin data for every lattice site. In the context of the Heisenberg model, the Hamiltonian for an individual atom $i$ is determined by the spin interaction with its neighbors:

$$H_i = f(s_i, s_j)$$

where $j$ represents the indices of adjacent lattice points. To enhance accuracy, the interaction range can be extended to include further neighbors, thereby capturing long-range effects within the lattice.

While traditional MCMC strictly adheres to sequential detailed balance, our approach relaxes these constraints to maximize parallel throughput. The procedure is outlined as follows:

1) Select a subset of $P$ distinct lattice points.
2) Propose a spin vector change for all $P$ points simultaneously.
3) Calculate the local energy change ($\Delta E$) for each point independently, assuming the neighbors are static (synchronous update).
4) Apply the Metropolis criterion to each point individually.
5) Store the acceptance result (YES $= 1$ or NO $= 0$) in a decision array $R$ of length $P$.
6) Execute all accepted updates in a single batch step.

During the evaluation phase, the initial state is effectively "locked in time." While this logic is compatible with sequential processing, it is optimized for parallel frameworks where each point in $P$ is assigned to a dedicated GPU core. This leverages the massive core count of modern GPUs to handle the lightweight per-site workload efficiently.

### B. Monte Carlo Step (MCS) and Parallelization Ratio

We introduce a metric termed the "Parallelization Ratio" ($P$), defined as the fraction of lattice sites selected for potential update in a single step relative to the total lattice size ($N$). In a classical MCMC, this corresponds to $P = 1/N$.

2770-0135/25/$31.00 ©2025 IEEE
DOI 10.1109/HiPCW66559.2025.00102

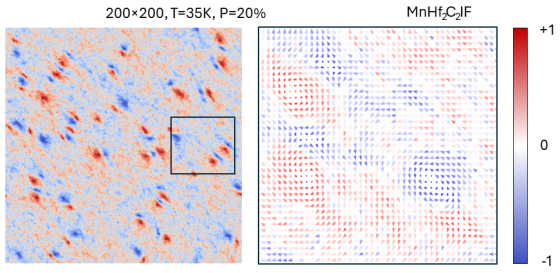


Fig. 1. Bipolar skyrmions in $MnHf_2C_2IF$. Color and intensity depict the direction and strength of the spin vector in the z-direction.

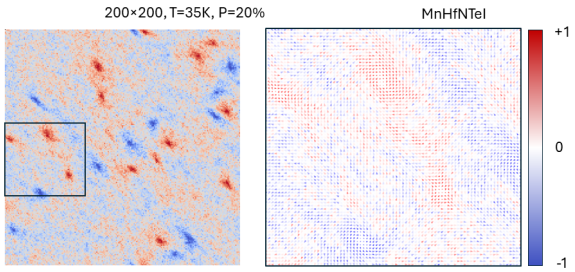


Fig. 2. Isolated directional skyrmions in MnHfNTeI. Color and intensity depict the direction and strength of the spin vector in the z-direction.

The parameter $P$ serves as a control variable for the simulation dynamics. Lower values of $P$ approximate true MCMC behavior, preserving detailed balance but suffering from slower convergence. Higher values of $P$ accelerate the simulation linearly but increase the deviation from exact detailed balance. The objective is to identify an optimal $P$ that balances computational speed with physical accuracy, allowing for rapid exploration of the phase space.

## III. Results

Building upon our initial validation, we present additional results regarding complex magnetic textures in various materials. For clarity, $P$ is expressed as a percentage.

We demonstrate the formation of advanced atomistic spin textures, specifically bipolar skyrmions in $MnHf_2C_2IF$ [5] and isolated skyrmions (both helicities) in MnHfNTeI [5]. The stability of these textures was cross-validated using Landau-Lifshitz-Gilbert (LLG) dynamics.

The high throughput of CUDA-METRO enables the observation of temporal phenomena, such as the complete lifecycle of skyrmions in MnSTe, as detailed in our previous study [3].

Furthermore, we explored the limits of the algorithm by scaling the lattice size. In the study of $VZr_3C_3II$ [5], we observed a critical transition in behavior as the lattice dimensions increased. Beyond a specific threshold, skyrmion formation ceases, and the lattice decomposes into two distinct stable domains. This phenomenon suggests that as the system

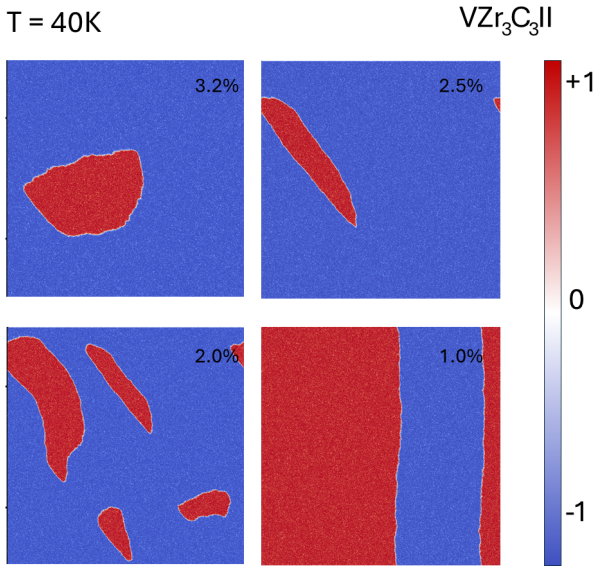


Fig. 3. Skyrmion formation in $VZr_3C_3II$ across various grid sizes ($N$). Above a critical threshold of $P$, microstructures form cleanly. However, for lattice sizes exceeding $1200 \times 1200$, the effective $P$ drops, preventing formation. Color and intensity depict the spin vector in the z-direction.

size grows, the effective $P$ relative to the total lattice size decreases. Consequently, the simulation mimics the behavior of traditional single-spin MCMC, becoming trapped in deep local minima rather than traversing the energy landscape effectively.

This limitation is tied to the hardware constraints (specifically, the number of available GPU cores). As lattice size ($N$) increases, the maximum achievable $P$ decreases. Since $P$ functions analogously to a "step length" in the energy landscape, a lower $P$ reduces the probability of the system making the large, asynchronous jumps required to escape local energy valleys. This finding highlights a significant weakness in existing high-accuracy MCMC methods: the tendency to prioritize local precision over global ergodicity. Our "unbound" algorithm provides a mechanism to overcome this by allowing aggressive sampling, making it uniquely interacting for analyzing magnetic megastructures in the micron regime.

## References

[1] A. Hait & S. Mahapatra, *Journal Of Open Source Software*. **10**, 7589 (2025,5), https://joss.theoj.org/papers/10.21105/joss.07589
[2] W Heisenberg *Zeitschrift For Physik*. **49**, 619-636 (1928,9)
[3] N. Metropolis et al. *The Journal Of Chemical Physics*. **21**, 1087-1092 (1953,6)
[4] NVIDIA, Vingelmann, P. & Fitzek, F. CUDA, release: 10.2.89. (2020), https://developer.nvidia.com/cuda-toolkit
[5] A. Kabiraj, & S. Mahapatra, *Npj Computational Materials*. **9**, 173 (2023,9)