Short-Range Order Based Ultra Fast Large-Scale Modeling of High-Entropy Alloys

Caimei Niu*, Lifeng Liu†
*School of Software and Microelectronics, Peking University
Email: caimeiniu@stu.pku.edu.cn
†School of Integrated Circuits, Peking University
Email: lfliu@pku.edu.cn

***Abstract*—High-Entropy Alloys (HEAs) exhibit complex atomic interactions, with short-range order (SRO) playing a critical role in determining their properties. Traditional methods, such as Monte Carlo (MC) and hybrid MC-Molecular Dynamics (MC-MD), are often constrained by system size and computational inefficiency, limiting their applicability for large-scale modeling. We introduce PyHEA, a high-performance Python package with a C++ core, designed to efficiently model HEAs through a combination of global and local search algorithms, incremental SRO calculations, and GPU acceleration. PyHEA delivers an impressive speedup of more than 110,000× over state-of-the-art tools like SCRAPs and ATAT, ensuring accurate SRO calculations with unprecedented efficiency. In practical applications, PyHEA successfully modeled a 256,000-atom Fe-Mn-Cr-Co alloy, replicating literature results and reducing computational time by over tenfold compared to hybrid MC-MD approaches. By eliminating the need for MC-MD hybridization, PyHEA reduces SRO simulation times from hours to minutes, offering a scalable and efficient tool for HEA research. This advancement enables unprecedented scale and efficiency in modeling atomic structures and SRO, accelerating HEA research and development.**

***Index Terms*—High-Entropy Alloys, Short-Range Order, High Performance Computing, Monte Carlo Search, Molecular Dynamics, GPU acceleration**

I. Introduction

High-Entropy Alloys (HEAs), introduced by Jien-Wei Yeh and colleagues in 2004 [1], represent a transformative class in materials science. Comprising five or more principal elements in near-equimolar ratios (typically 5–35 atomic percent each), HEAs exhibit a suite of remarkable properties, including high strength, hardness, and exceptional resistance to oxidation and corrosion [2–5]. These properties make HEAs promising candidates for applications in extreme environments, such as aerospace, nuclear, and energy sectors, while also challenging foundational theories in alloy design [3,4,6,7].

Recent research has revealed the importance of microstructure, especially short-range order (SRO), in determining HEAs' unique properties [6,8]. particularly short-range order (SRO), which refers to the local atomic arrangement within an alloy [9–13]. SRO, describing the local atomic arrangement, emerges as a prevalent feature in complex concentrated alloys, even under rapid solidification [14]. It significantly impacts mechanical performance and thermal stability, enhancing elastic modulus, stiffness, and, in some cases, ductility through an optimized balance of strength and toughness [6,13,15–17]. These findings suggest a potential for precise SRO control in tailoring HEA properties, laying a theoretical foundation for new material development.

However, leveraging SRO in HEA design remains a substantial challenge due to the exponentially large configuration space resulting from multiple principal elements [18–20]. For example, in a 250-atom equiatomic Cantor alloy (CrMnFeCoNi), the configuration space is approximately $\frac{250!}{(50!)^5} \approx 10^{170}$ — a complexity that escalates with increasing atom count.

Current methods for modeling SRO in HEAs face scalability limitations [21]. Simulation tools like ATAT [22,23] and SCRAPs [24], while effective for small to medium-scale systems, encounter efficiency constraints in larger systems and struggle to capture SRO effects, particularly under low-temperature conditions where SRO becomes critical [25–27]. As system size increases, these methods often become computationally prohibitive, limiting their ability to simulate HEAs' comprehensive behavior at macroscopic scales.

Hybrid Monte Carlo-Molecular Dynamics (MC-MD) methods have been proposed for simulating larger systems of hundreds of thousands of atoms [18,28,29]. Although these methods support extensive simulations, they demand significant computational resources and rely on highly accurate interatomic potentials, derived from Density Functional Theory (DFT) or Machine Learning Interatomic Potentials (MLIPs) [30–33]. This dependency further exacerbates computational costs, underscoring the need for an efficient modeling framework capable of handling large HEA systems with specific SRO requirements without prohibitive computational demands.

To address these challenges, we introduce PyHEA,

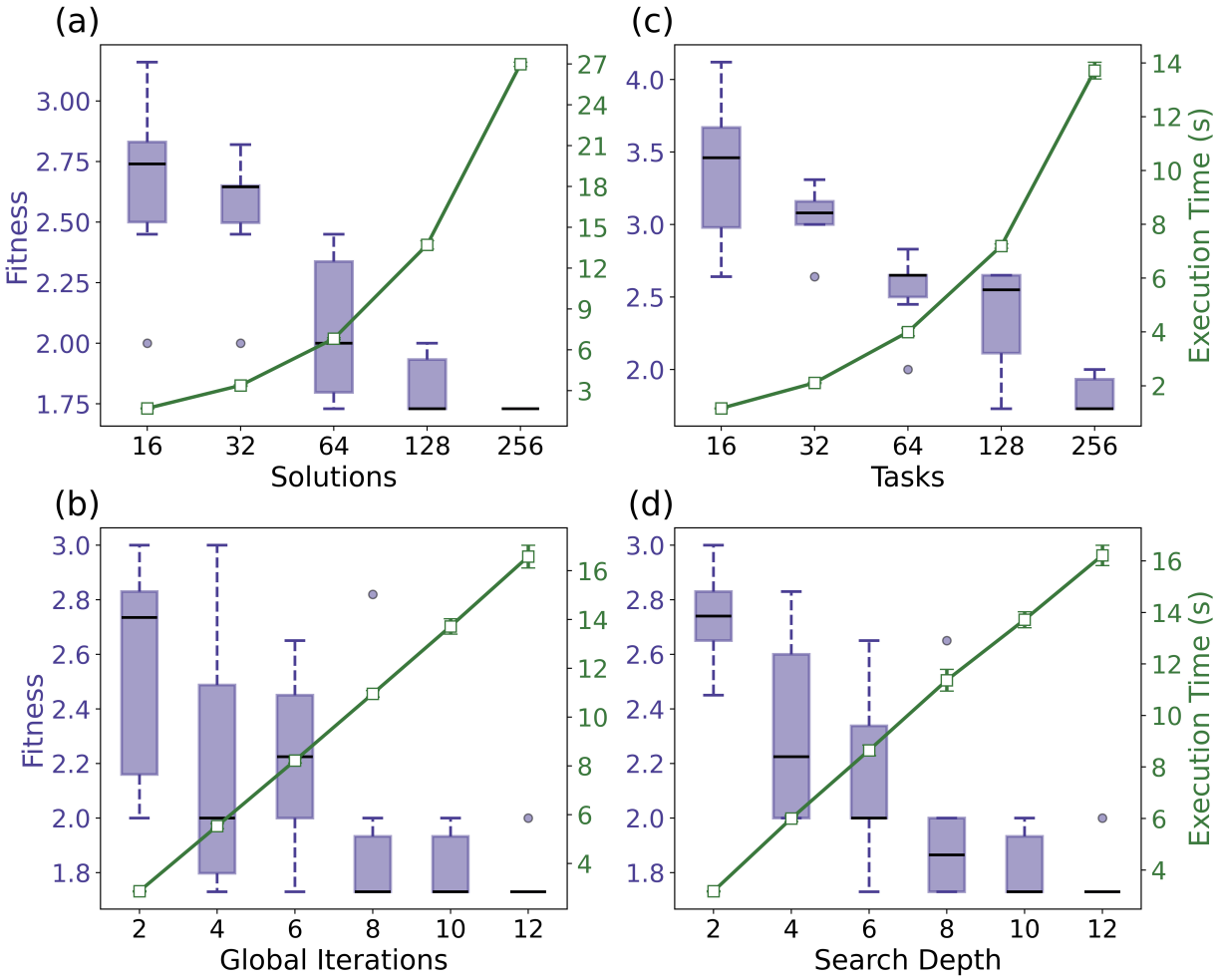

Fig. 1: Influence of input parameters on fitness and execution time. (a) Number of solutions, (b) global iterations, (c) number of tasks, and (d) search depth. The plots illustrate the impact of varying each parameter on the fitness (left axis) and execution time (right axis).

a computational framework optimized for large-scale HEA modeling with two core innovations aimed at overcoming scalability and efficiency issues in capturing SRO. First, PyHEA combines global and local search algorithms, which balances broad exploration with detailed refinement, efficiently navigating the expansive configuration space typical in HEAs and significantly reducing modeling times. Second, it leverages high-precision potentials, such as those from DFT and MLIPs, within an incremental SRO calculation framework. This approach allows PyHEA to rapidly evolve SRO values with precision, enabling the construction of accurate, large-scale HEA models within minutes rather than days. By harnessing both robust search techniques and precise local interactions, PyHEA advances HEA modeling capabilities, allowing researchers to conduct simulations at unprecedented scales and detail.

II. Results

A. Parameter Optimization Analysis

This section details the key input parameters and their roles in the optimization process. These parameters include the number of solutions, global iterations, tasks, and search depth, all of which directly influence the balance between computational cost and the quality of the fitness function, defined here as the degree to which the simulated atomic configurations align with target SRO characteristics.

In this context, the fitness function serves as a metric for evaluating the alignment of atomic configurations with the desired SRO properties, where higher fitness values indicate more optimized structures. As illustrated in Fig. 1, the parameters—comprising the number of solutions, global iterations, tasks, and search

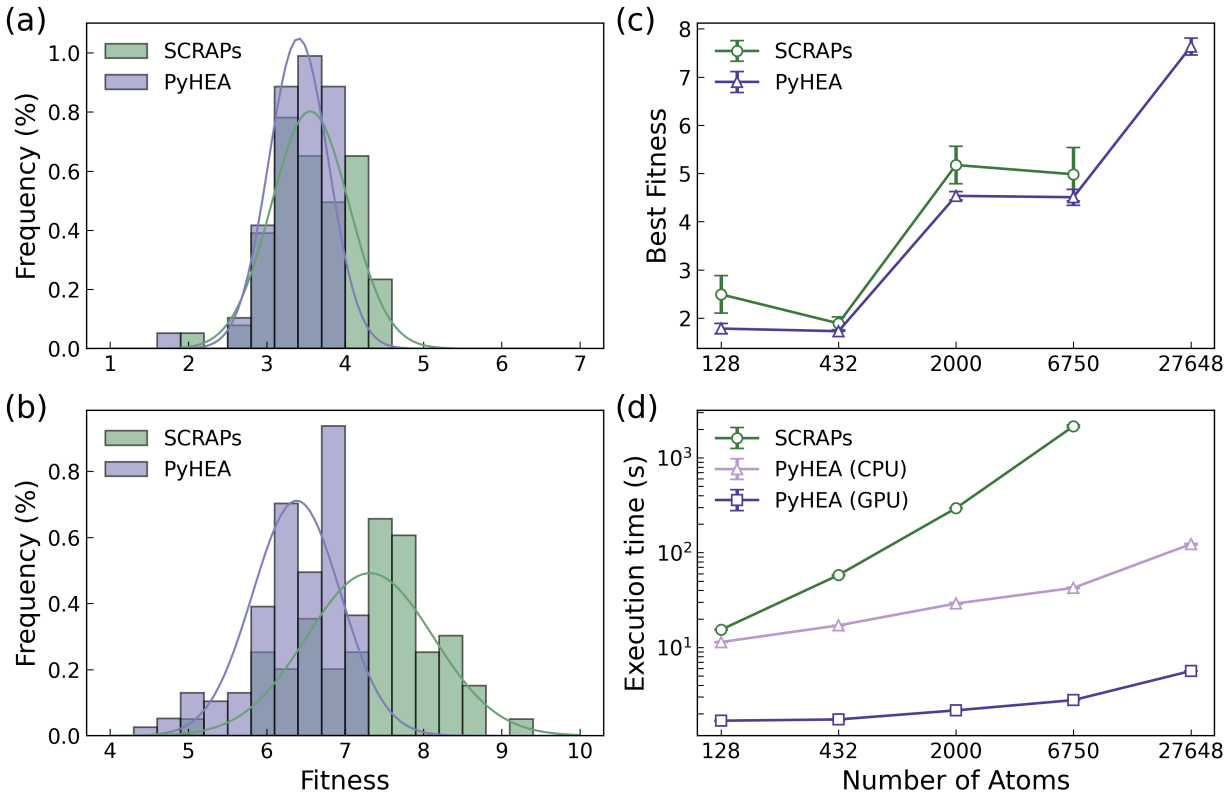

Fig. 2: Comparison of PyHEA and SCRAPs in terms of fitness accuracy and execution time. (a) and (b) show the distribution of fitness values (lower values indicate higher accuracy) for systems containing 128 atoms and 2000 atoms, respectively. (c) illustrates the best fitness values as a function of system size. (d) presents the execution times for SCRAPs, PyHEA (CPU), and PyHEA (GPU) across various system sizes. Both SCRAPs and PyHEA (CPU) were tested on the same CPU cluster equipped with an Intel Xeon Gold 6132 CPU @ 2.60 GHz using 24 processors, while PyHEA (GPU) utilized an NVIDIA 4090 GPU.

depth—significantly affect the trade-off between the quality of fitness and computational efficiency.

As depicted in Fig. 1(a-b), an increase in the number of solutions improves fitness; however, beyond 64 solutions, this enhancement yields diminishing returns while causing a marked rise in execution time. A similar pattern emerges for global iterations, where substantial fitness improvements are observed up to 6 iterations. Beyond this threshold, further gains are minimal, whereas execution time continues to escalate. These results indicate that using 64 to 128 solutions and 6 global iterations achieves an effective balance between precision and computational cost.

Parallelism also plays a critical role in the optimization process. As demonstrated in Fig. 1(c), increasing the number of tasks enhances the consistency of fitness outcomes. Nevertheless, execution time grows significantly after 128 tasks, with only marginal improvements in fitness. Therefore, a task count between 64 and 128 represents an optimal range. Similarly, search depth affects the thoroughness of exploration within each iteration. Although deeper searches (greater than 6) yield better fitness outcomes, they come at the expense of increased computational demands. Based on these findings, a search depth of 4 to 6 strikes an optimal balance between precision and computational efficiency.

Consequently, the recommended parameter set for this optimization process comprises 64-128 solutions, 6 global iterations, 64-128 tasks, and a search depth of 4-6. This configuration ensures scalability and computational feasibility while maintaining high precision in fitness assessments.

B. Comparison with State-of-the-Art Methods

To assess the performance of PyHEA, we conducted a comparative analysis against SCRAPs[24], a state-of-the-art method commonly utilized in HEA modeling. This comparison focused on evaluating both fitness quality and computational efficiency across varying system sizes.

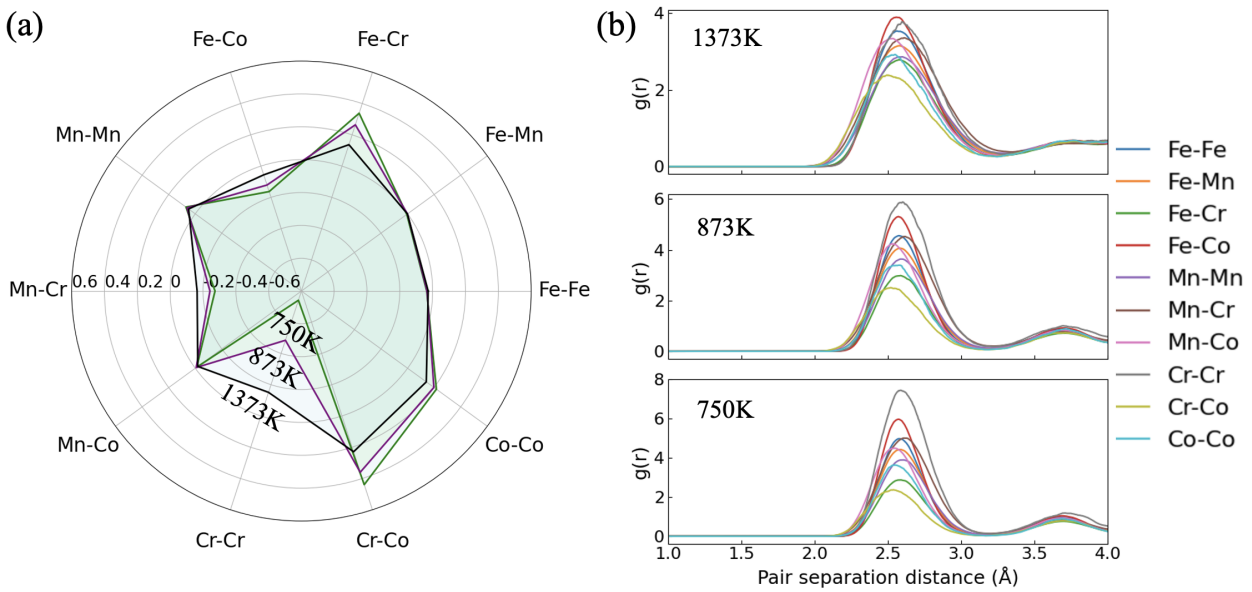

Fig. 3: Combined visualization of SRO and radial distribution functions (RDF) for atomic pair interactions at different temperatures. (a) The radar plot illustrates the SRO values for key atomic pairs (e.g., Fe-Fe, Fe-Mn, Fe-Cr) at three temperatures: 750 K, 875 K, and 1373 K. (b) The RDF plots display the spatial distribution of atomic pairs as a function of separation distance (Å) for the same temperatures, with smooth curves representing each pair.

Figs. 2(a) and 2(b) present the fitness distributions for systems with 128 atoms and 2000 atoms, respectively. PyHEA consistently exhibits a more concentrated fitness distribution compared to SCRAPs, particularly as the system size increases. In the case of the 2000-atom system, PyHEA demonstrates a narrower spread with a higher peak frequency around the optimal fitness value, signifying more stable and reliable outcomes. Conversely, SCRAPs shows a broader distribution of fitness values, indicating that PyHEA's optimization strategy is more robust, especially in larger systems.

The enhanced fitness quality of PyHEA is further corroborated by Fig. 2(c), which tracks the optimal fitness values as system size increases. While PyHEA and SCRAPs achieve comparable fitness levels for smaller systems (e.g., 128 and 432 atoms), PyHEA surpasses SCRAPs as the system size exceeds 2000 atoms, highlighting its scalability and the effectiveness of its optimization approach. This is particularly crucial for large-scale HEA simulations where fitness convergence is increasingly challenging.

A key advantage of PyHEA is its computational efficiency, particularly with GPU acceleration. Fig. 2(d) compares the execution times of PyHEA (in both CPU and GPU implementations) and SCRAPs as a function of system size. SCRAPs faces significant scalability challenges, with execution times increasing rapidly as the system size grows. Notably, SCRAPs becomes computationally prohibitive at approximately 6750 atoms, beyond which further calculations are unfeasible.

In contrast, PyHEA's GPU implementation seamlessly handles larger system sizes. At the largest system size of 27,648 atoms, PyHEA's GPU-accelerated version achieves a performance improvement of four orders of magnitude compared to SCRAPs. This significant boost in computational efficiency allows PyHEA to handle system sizes that are beyond the practical capacity of SCRAPs.

To further validate PyHEA's acceleration benefits, we also conducted a comparison with another widely used MC-based tool, ATAT. Table I summarizes the performance metrics across PyHEA (CPU and GPU), SCRAPs, and ATAT. While SCRAPs and ATAT perform comparably for smaller system sizes, they both struggle as the number of atoms increases. Like SCRAPs, ATAT becomes impractical beyond 10,000 atoms. Conversely, PyHEA's GPU implementation maintains substantially lower execution times across all tested system sizes, underscoring its performance advantages and scalability.

C. Application

To illustrate the practical potential of PyHEA, we conducted large-scale molecular dynamics (MD) simulations using LAMMPS, modeling a system containing 256,000 atoms of Fe-Mn-Cr-Co (composition: 40% Fe, 40% Mn, 10% Cr, and 10% Co) at various temperatures. The system was generated using PyHEA, enabling us

TABLE I: PyHEA computation times (in minutes) for generating optimal structures using different hardware configurations, demonstrating speedups over state-of-the-art methods. The search space consisted of 128 solutions, with the global search running for 10 iterations (typically converging within 3 to 5 iterations). Each iteration employed up to 256 parallel MC tasks, each with a search depth of 10. For simplicity, supercell sizes were set as $S = A \cdot L^3$, where L denotes the length of the cubic supercell, and A = 2 for body-centered cubic (bcc) or A = 4 for face-centered cubic (fcc), representing the number of atoms in the primitive cell. Although shells can be included to a range that avoids correlation from periodic boundaries, the short-range order (SRO) was optimized over 3 shells. Comparisons are made with popular MC-based ATAT code[23] and the SCRAPs method[24]. Calculations used a cluster (Intel(R) Xeon(R) Gold 6132 CPU @ 2.60GHz/28-threads) and a NVIDIA 4090 GPU.

Type	Species	Atoms	ATAT	SCRAPs serial [24 cores]	PyHEA CPU serial [24 cores]	PyHEA GPU	CPU speedup over SCRAPs [ATAT]	GPU speedup over SCRAPs [ATAT]
bcc	3	54	>1440[24]	2.74[0.14]	3.22[0.14]	0.028	0.85[447]	98[51429]
bcc	4	128	>10,000[24]	6.51[0.34]	4.19[0.19]	0.028	1.55[2387]	233[357143]
bcc	4	432	-	24.22[1.26]	5.62[0.28]	0.029	4.31	835
bcc	4	1024	-	103.14[3.18]	6.24[0.32]	0.032	16.53	3223
bcc	4	2000	-	276.00[7.02]	7.00[0.36]	0.035	39.43	7886
bcc	5	2000	-	285.48[7.27]	9.28[0.48]	0.036	30.11	7930
bcc	5	6750	-	639.45[47.44]	14.37[0.71]	0.046	44.50	13901
bcc	6	27648	-	-	39.61[2.05]	0.092	-	-
fcc	3	108	-	17.37[0.63]	3.89[0.17]	0.005	4.47	3474
fcc	4	256	-	68.77[2.24]	5.50[0.27]	0.006	12.50	11461
fcc	5	500	-	260.16[5.81]	9.39[0.43]	0.007	27.71	37166
fcc	5	4000	-	1559.2[64.31]	14.17[0.65]	0.014	110.0	111370
fcc	4	256000	-	-	1167.2[61.08]	0.577	-	-

[a] Due to excessive computational demands, multinary results cannot be provided.
[b] The CPU speedup ratio tests is based on the serial results.

to bypass the conventional Monte Carlo (MC) and MD hybrid process and achieve significant reductions in simulation time.

The system was simulated in an NPT ensemble for 10 ps, with the Radial Distribution Function (RDF) calculated to analyze the SRO behavior at different temperatures. As depicted in Fig. 3, our results align closely with those from a reference study in which MC simulations were performed on a comparable Fe-Mn-Cr-Co system to investigate compositionally-driven SRO[34]. Additionally, our simulation outcomes were consistent with the ordering patterns reported in experimental studies[35,36] which highlights the Fe–Mn/Co/Cr preference in the first coordination shell within an fcc $Fe_{50}Mn_{30}Cr_{10}Co_{10}$ system.

Employing PyHEA to generate input structures for LAMMPS allowed us to circumvent the extensive MC-MD hybrid computations typically required, which can take several days to complete. PyHEA reduced this computation time to just a few minutes, showcasing a remarkable improvement in computational efficiency. Furthermore, the scale of the simulation, involving more than 200,000 atoms, surpasses the capabilities of most existing methods, emphasizing PyHEA's robustness and effectiveness in handling large-scale atomistic simulations.

III. Discussion

This study demonstrates that PyHEA achieves superior accuracy and can efficiently handle significantly larger system sizes compared to traditional methods. While existing tools such as SCRAPs and ATAT show exponential increases in execution time for systems exceeding 6750 atoms and LAMMPS encounters substantial computational bottlenecks, PyHEA's GPU acceleration supports simulations with over 200,000 atoms (Table I), delivering performance improvements of up to four orders of magnitude. This scalability is critical for investigating material properties at more realistic scales.

A notable achievement of this work is PyHEA's ability to substantially reduce the time required for SRO simulations. Traditional approaches utilizing LAMMPS or hybrid MC-MD methods often require several days to complete; in contrast, PyHEA's optimized workflow reduces simulation time to just a few minutes without sacrificing accuracy (Fig. 5). This level of efficiency not only allows for the simulation of extensive system modeling but also ensures accurate capture of complex atomic interactions and SRO.

In practical applications, PyHEA successfully simulated intricate SRO phenomena in a 256,000-atom Fe-Mn-Cr-Co system across various temperatures. The consistency of PyHEA's RDF results with those reported in existing literature further supports its accuracy and applicability in real-world scenarios. This underscores PyHEA's value not only as a computational tool but also

as a reliable method for conducting HEA simulations. The ability to conduct simulations at this scale opens the door for future research into more complex, multi-component alloy systems, potentially impacting other fields of materials science and alloy design.

PyHEA's scalability and performance provide broader implications for the research community. The reduction in simulation time from days to minutes facilitates faster research cycles and reduces the computational resources needed, making large-scale atomistic simulations more feasible and cost-effective, even for teams with limited resources. This capability supports more iterative and exploratory research, where rapid adjustments to simulation parameters and re-analysis can lead to deeper insights and faster progress.

In conclusion, PyHEA represents a significant advancement in HEA modeling by addressing the computational challenges associated with large-scale simulations. By integrating global and local search algorithms, GPU optimization, and incremental SRO calculations, PyHEA surpasses state-of-the-art methods such as SCRAPs, ATAT, and LAMMPS in both accuracy and scalability. Its ability to handle systems with over 200,000 atoms while maintaining computational efficiency establishes it as a leading tool for HEA research. The reduction in simulation time, along with consistent fitness evaluations, facilitates a more comprehensive exploration of material properties, particularly the role of SRO in HEA performance.

Future research could focus on further optimizing PyHEA's algorithms and expanding their application to larger and more complex material systems. Incorporating advanced machine learning models, such as reinforcement learning, could enhance PyHEA's adaptive capabilities in optimizing SRO calculations. Additionally, collaborative efforts with experimental teams to validate PyHEA's predictions under different thermomechanical conditions would strengthen its applicability and predictive accuracy. Ultimately, PyHEA signifies a major step forward in HEA modeling, providing a scalable, accurate, and efficient platform for simulating short-range order and atomic interactions in extensive systems.

IV. Methods

A. Global Search

As shown in Fig. 4(a), the global search phase is essential for navigating the extensive configuration space of atomic arrangements in HEAs. This phase strikes a balance between exploration and refinement, allowing for the identification of promising regions within the solution space.

The search process begins by generating candidate solutions that represent different atomic configurations (Fig. 4(b)). During each iteration, a subset of solutions undergoes random perturbations to enable broader exploration, while the top-performing configurations are preserved for further refinement (Fig. 4(d)). This approach effectively integrates random exploration with solution improvement.

Algorithm 1 Global Search Algorithm

Require: Fix input and optimization function
Ensure: Optimized solution
1: Initialize solutions
2: **while** iteration $<$ Global maximum number **do**
3: Create new solutions randomly
4: Calculate fitness F of the solutions
5: Choose fraction of solutions with best fitness (top solutions)
6: **for** each solution $\in$ top solutions **do**
7: Search using Monte Carlo (Local Search)
8: **end for**
9: Discard fraction p_a of worst solutions
10: Rank the solutions and find the current best
11: **end while**
12: Return the best solutions

Drawing inspiration from the Cuckoo Search algorithm, the method employs L'evy flights for optimization[24,37]. These random jumps facilitate global exploration and help prevent the algorithm from becoming trapped in local minima. Each candidate solution is evaluated using a fitness function, $F(X)$, which quantifies the quality of SRO in the atomic structure by comparing local atomic arrangements to the desired SRO characteristics.

To maintain solution diversity, a portion of low-performing solutions is eliminated at the end of each iteration, while the best solutions are carried forward to the next cycle. This iterative procedure continues until convergence, gradually driving the system toward an optimized atomic configuration with enhanced SRO properties.

B. Local Search

The local search phase plays a critical role in refining candidate solutions obtained from the global search phase, particularly in large-scale HEA simulations. To achieve high performance and precision, a parallelized Monte Carlo (MC) search (Algorithm 3)is employed, leveraging GPU implementation to efficiently refine atomic configurations in parallel. This parallelization enables the distribution of tasks across multiple threads, where configurations are perturbed, fitness differences are evaluated, and only the highest-performing solutions are retained. This method enhances both computational

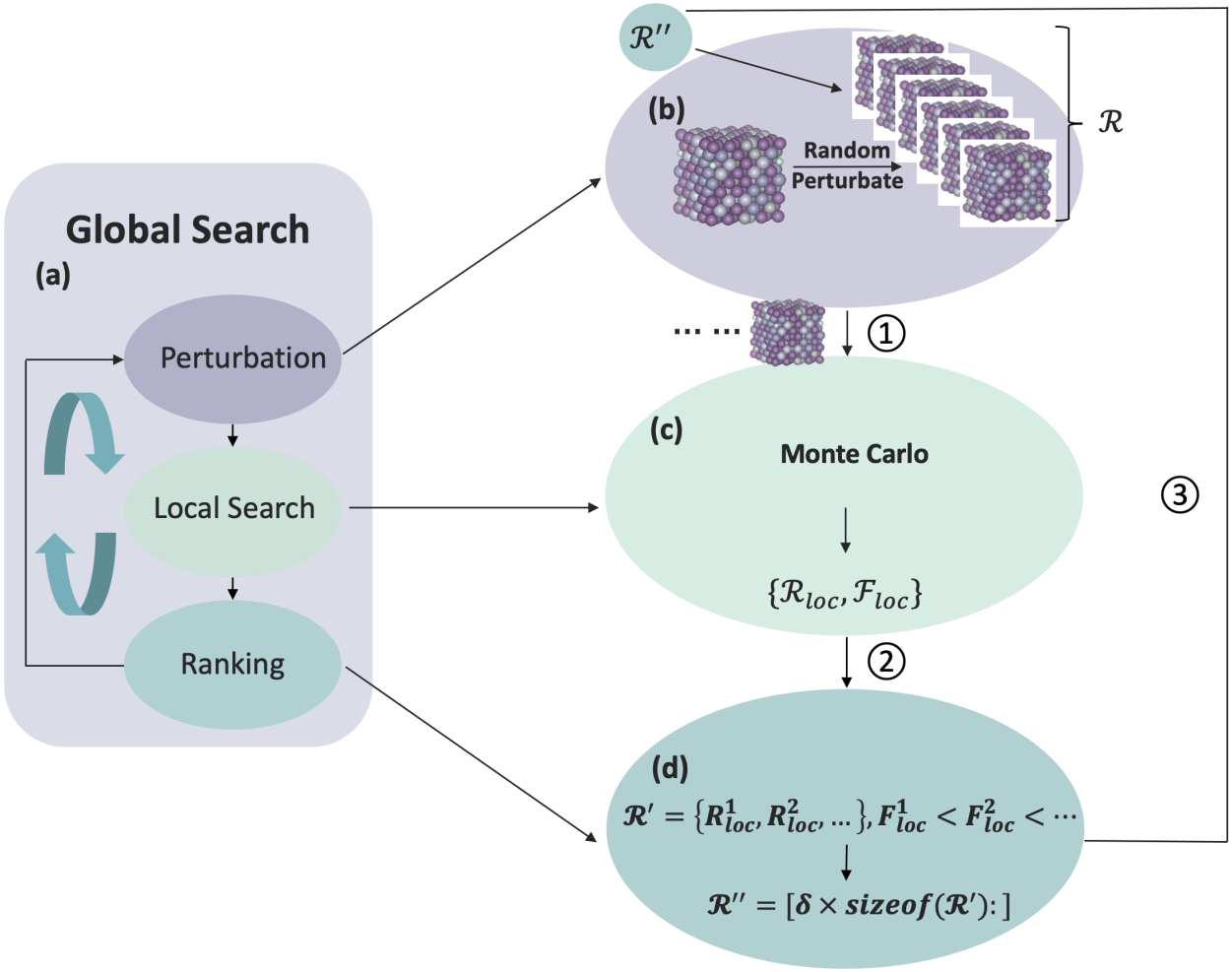

Fig. 4: Workflow of the PyHEA algorithm for optimizing atomic configurations. (a) The global search framework iteratively cycles through the stages of perturbation, local search, and ranking to converge toward an optimal solution. (b) Initial atomic configurations consist of randomly generated and previously identified optimal structures for exploration. (c) Monte Carlo (MC) methods are applied to refine each candidate configuration and evaluate their fitness. (d) Configurations are ranked based on fitness, and the lowest-ranked fraction of solutions is discarded to enhance the selection process for subsequent iterations.

Algorithm 2 Serial Monte Carlo Search Algorithm

Require: Fix input and optimization function
Ensure: Optimized solution
1: Initialize current configuration
2: **while** iteration $<$ Local iterations **do**
3: Perturb configuration randomly
4: Calculate fitness difference $\delta F = F(x) - F(x_{\text{new}})$
5: **if** $\delta F < 0$ **or** $|\delta F| \geq$ threshold **then**
6: Restore fitness and configuration to previous state
7: **end if**
8: **end while**
9: Return the best solutions

speed and accuracy, significantly reducing the overall computation time.

Compared to the serial version (Algorithm 2), which processes configurations one at a time and struggles with the scalability required for large systems, the parallel approach addresses this limitation by processing multiple configurations simultaneously. This parallelization significantly enhances both scalability and performance. By integrating the parallelized local search with the global search phase, candidate solutions are refined to achieve optimal local configurations, facilitating convergence to high-quality results with improved SRO properties.

Algorithm 3 Parallel Monte Carlo Search Algorithm

Require: Fix input and optimization function
Ensure: Optimized solution
Initialize current configuration
for each task i in parallel **do**
    Load current fitness and configuration
    Perturb configuration randomly
    Calculate fitness difference $\delta F_i = F(x) - F(x_i)$
    Synchronize tasks and sort δF_i in descending order
    if $\delta F_{min} > 0$ **or** $|\delta F_{min}| \leq$ threshold **then**
        $depth$ = 0
        Update current fitness and configuration to x_i
    else
        $depth$ += 1
    end if
    if $depth \geq$ threshold **then**
        Break
    end if
end for
Return the best solutions

C. Short-Range Order

Short-Range Order (SRO) characterizes the local atomic structure in HEAs, describing the tendency of certain atomic species to preferentially cluster compared to a random distribution[38]. The SRO is typically quantified by the Warren-Cowley parameter[9], denoted as α_{ij}^k, which measures the degree of ordering between atomic species i and j. For a given atom i, the SRO parameter for its k-th coordination shell is defined as:

$$\alpha_{ij}^k = 1 - \frac{P_{ij}^k}{c_j}, \tag{1}$$

where $P_{ij}^{(k)}$ is the conditional probability of finding a j atom in the k-th coordination shell of an i atom, c_j is the overall concentration of j atoms in the alloy.

The SRO parameter α_{ij}^k ranges from $[-1, 1]$:

- $\alpha_{ij}^k < 0$: Indicates an attraction between elements i and j, suggesting a preference to be adjacent.
- $\alpha_{ij}^k > 0$: Indicates a repulsion between elements i and j, suggesting a preference to be separated.

For each atomic pair type (i, j), the SRO parameter for the entire system of N atoms can be expressed as:

$$\langle \alpha_{ij}^k \rangle = \frac{1}{N_i} \sum_{n=1}^{N_i} \left(1 - \frac{P_{ij}^k}{c_j}\right), \tag{2}$$

where N_i is the number of atoms of type i.

To calculate this parameter, the number of j atoms within each coordination shell around every i atom is determined, and the result is normalized by the total concentration of j atoms (c_j). computed SRO values for each pair (i, j) are then averaged over the entire system to obtain a global measure of the SRO.

D. Molecular Dynamics

Molecular dynamics (MD) simulations were performed using LAMMPS[39–41], employing the 2-NN MEAM potential for the Fe-Mn-Cr-Co alloy system. The simulations encompassed system sizes ranging from 256 to 260,000 atoms, with a timestep of 2 fs in an NPT ensemble. The built-in MC method in LAMMPS was utilized to compute the Radial Distribution Function (RDF), providing detailed insight into the atomic arrangement and SRO.

These simulation parameters were chosen to ensure reproducibility and to facilitate replication in future studies. The results of these MD simulations are discussed in detail in Section II, demonstrating the effectiveness of our method.

E. Incremental Calculation of SRO (Fitness)

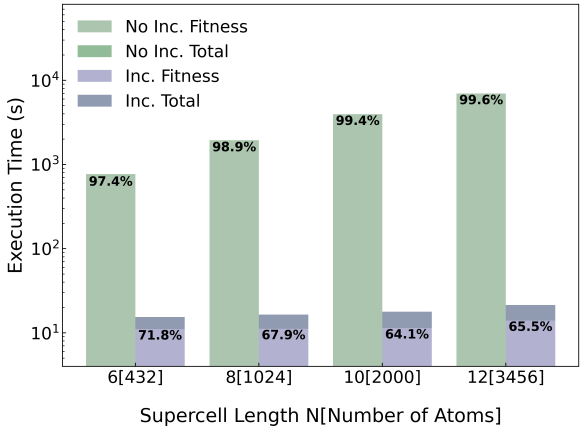

Fig. 5: Comparison of execution times for incremental and non-incremental fitness and total calculations in Monte Carlo search. The bar chart shows the execution times (in seconds) for different supercell lengths N and corresponding atom counts. Incremental methods demonstrate substantial improvements, reducing execution time for both fitness evaluation and total computation. The incremental approach achieves up to a 99.6% reduction in total computation time for larger supercells.

In the Monte Carlo (MC) search process, the SRO parameter is crucial for evaluating the fitness of each atomic configuration. As previously discussed, SRO quantifies the local atomic arrangement, guiding the search toward optimal configurations during each MC iteration.

In the traditional non-incremental approach, SRO is recalculated from scratch at every MC iteration (Eqs. 2). This recalculation becomes increasingly computationally expensive as system size grows. Fig. 5 illustrates that for larger systems, the SRO calculation can consume more than 99% of the total computational cost, highlighting the necessity for a more efficient method.

To address this challenge, we implemented an incremental SRO calculation. Instead of recalculating the SRO for the entire system at each iteration, we update only the SRO values influenced by the random swap of two atoms, i and j. This approach localizes the update to the environments of the swapped atoms and their neighbors, enabling a more efficient adjustment of the SRO during each MC iteration.

For the incremental update, we first compute the change in SRO for the atoms directly involved in the swap. The change in the SRO value for atom i and its neighbors is given by:

$$\Delta\alpha_{ij}^{k} = \left(1 - \frac{P_{ij}^{k}(\text{new})}{c_j}\right) - \left(1 - \frac{P_{ij}^{k}(\text{old})}{c_j}\right), \quad (3)$$

where P_{ij}^{k}(new) refers to the probability of finding a j atom in the updated configuration after the swap, and P_{ij}^{k}(old) is the corresponding probability before the swap. This incremental approach reduces the computational complexity of each MC iteration from $O(N)$ to $O(1)$, as only the swapped atoms and their neighboring atoms are updated. This improvement is essential for scaling simulations to systems containing hundreds of thousands of atoms without overwhelming computational resources.

Benchmark data shown in Fig. 5 highlights the significant computational burden of non-incremental SRO calculations, which account for 97.4% to 99.6% of the total execution time as system sizes increase from 432 to 3,456 atoms. However, the incremental SRO approach substantially alleviates this burden, reducing total execution times to 71.8% for 432 atoms and stabilizing around 65.5% for larger systems. These findings underscore the incremental method's ability to enhance computational efficiency and scalability for large-scale HEA simulations.

By adopting incremental SRO updates, we maintain the accuracy of fitness evaluations while significantly reducing the computation time for each MC iteration. This ensures that the MC search process remains efficient and scalable for large-scale HEA simulations.

F. GPU Optimization

To further accelerate the MC search and enhance scalability, we implemented a parallel GPU-based optimization. Each thread processes an independent task, and each thread block manages the complete MC search for a single solution. This approach allows the GPU to handle multiple solutions simultaneously, leveraging resources such as registers, shared memory, and low-precision data formats to maximize computational efficiency.

1) Parallel Processing on the GPU: In this setup, each thread performs the following tasks:

- **SRO Calculation**: Threads compute the SRO for their assigned atomic configurations using low-precision formats (FP16, Uint16) to reduce data footprint and memory access latency, while maintaining sufficient accuracy.
- **Lattice Structure Storage**: The lattice structure[42] is stored in shared memory to minimize memory access time during each MC iteration. This allows all threads within a block to efficiently access and update atomic positions, enhancing computational speed.derived from the SRO values calculated in registers, is updated following each perturbation:

2) Iterative Fitness Calculation: Each thread in the block computes the fitness of the perturbed solution during every MC iteration. The fitness, derived from the SRO values calculated in registers, is updated following each perturbation:

$$F_{\text{new}} = F_{\text{prev}} + \Delta F_{\text{perturb}}, \quad (4)$$

where $\Delta F_{\text{perturb}}$ represents the fitness change from the most recent atomic position perturbation. This value is calculated using the incremental SRO update method.

3) Fitness Sorting and Selection: After computing the updated fitness in registers, threads within the block collaborate to sort their fitness values using shared memory. The sorting process ensures the rapid identification of the best-performing configuration, which updates the current optimal solution stored in shared memory:

$$F_{\text{best}} = \min\left(F_{\text{new}}\right) \quad (5)$$

This guarantees that each block retains the best solution found during the current iteration.

4) Convergence Check and Termination: The MC search continues iterating until convergence is achieved. Convergence is defined as the point where the fitness of the current best solution shows no significant improvement over a fixed number of steps. Specifically, if $\Delta F_{\text{best}} = 0$ for a predefined threshold of iterations, the threads within the block terminate the search. The final result, including the atomic configuration and the associated fitness, is then written back to global GPU memory for subsequent analysis.

This GPU-based approach significantly accelerates the MC search by distributing tasks across multiple threads and blocks, enabling the efficient handling of large-scale systems.

V. Acknowledgment

This work was conducted under the supervision of Professor Lifeng Liu, School of Integrated Circuits,, Peking University. The author acknowledges the financial support provided by the National Natural Science Foundation of China (Grant No. 61874006).

VI. Code Availability

The code used for this study is available at https://github.com/caimeiniu/pyhea. Access is granted under LGPL-3.0 license.

References

[1] J.-W. Yeh, S.-K. Chen, S.-J. Lin, J.-Y. Gan, T.-S. Chin, T.-T. Shun, C.-H. Tsau, S.-Y. Chang, Nanostructured high-entropy alloys with multiple principal elements: novel alloy design concepts and outcomes, Advanced engineering materials 6 (5) (2004) 299–303.

[2] D. B. Miracle, O. N. Senkov, A critical review of high entropy alloys and related concepts, Acta materialia 122 (2017) 448–511.

[3] E. P. George, D. Raabe, R. O. Ritchie, High-entropy alloys, Nature reviews materials 4 (8) (2019) 515–534.

[4] W. Li, D. Xie, D. Li, Y. Zhang, Y. Gao, P. K. Liaw, Mechanical behavior of high-entropy alloys, Progress in Materials Science 118 (2021) 100777.

[5] J. W. Yeh, Y. L. Chen, S. J. Lin, S. K. Chen, High-entropy alloys–a new era of exploitation, in: Materials science forum, Vol. 560, Trans Tech Publ, 2007, pp. 1–9.

[6] Y. Zhang, T. T. Zuo, Z. Tang, M. C. Gao, K. A. Dahmen, P. K. Liaw, Z. P. Lu, Microstructures and properties of high-entropy alloys, Progress in materials science 61 (2014) 1–93.

[7] Z. Li, K. G. Pradeep, Y. Deng, D. Raabe, C. C. Tasan, Metastable high-entropy dual-phase alloys overcome the strength–ductility trade-off, Nature 534 (7606) (2016) 227–230.

[8] Y. Zhang, X. Yang, P. Liaw, Alloy design and properties optimization of high-entropy alloys, Jom 64 (2012) 830–838.

[9] J. M. Cowley, An approximate theory of order in alloys, Physical Review 77 (5) (1950) 669.

[10] D. De Fontaine, Cluster approach to order-disorder transformations in alloys, in: Solid state physics, Vol. 47, Elsevier, 1994, pp. 33–176.

[11] Q. Ding, Y. Zhang, X. Chen, X. Fu, D. Chen, S. Chen, L. Gu, F. Wei, H. Bei, Y. Gao, et al., Tuning element distribution, structure and properties by composition in high-entropy alloys, Nature 574 (7777) (2019) 223–227.

[12] R. Zhang, S. Zhao, J. Ding, Y. Chong, T. Jia, C. Ophus, M. Asta, R. O. Ritchie, A. M. Minor, Short-range order and its impact on the crconi medium-entropy alloy, Nature 581 (7808) (2020) 283–287.

[13] T. Abe, Effect of short-range ordering in high-entropy alloys, Materials transactions 62 (6) (2021) 711–718.

[14] Y. Han, H. Chen, Y. Sun, J. Liu, S. Wei, B. Xie, Z. Zhang, Y. Zhu, M. Li, J. Yang, et al., Ubiquitous short-range order in multi-principal element alloys, Nature Communications 15 (1) (2024) 6486.

[15] E. Pickering, N. Jones, High-entropy alloys: a critical assessment of their founding principles and future prospects, International Materials Reviews 61 (3) (2016) 183–202.

[16] S. Chen, Z. H. Aitken, S. Pattamatta, Z. Wu, Z. G. Yu, D. J. Srolovitz, P. K. Liaw, Y.-W. Zhang, Simultaneously enhancing the ultimate strength and ductility of high-entropy alloys via short-range ordering, Nature communications 12 (1) (2021) 4953.

[17] X. Liu, W. Curtin, Atomistic simulations reveal strength reductions due to short-range order in alloys, Acta Materialia 263 (2024) 119471.

[18] A. Ferrari, F. Körmann, M. Asta, J. Neugebauer, Simulating short-range order in compositionally complex materials, Nature Computational Science 3 (3) (2023) 221–229.

[19] C. Zhang, Y. Yang, The calphad approach for heas: Challenges and opportunities, MRS Bulletin 47 (2) (2022) 158–167.

[20] J.-W. Yeh, Alloy design strategies and future trends in high-entropy alloys, Jom 65 (2013) 1759–1771.

[21] M. He, W. J. Davids, A. J. Breen, S. P. Ringer, Quantifying short-range order using atom probe tomography, Nature Materials 23 (9) (2024) 1200–1207.

[22] G. Ghosh, A. Van de Walle, M. Asta, First-principles calculations of the structural and thermodynamic properties of bcc, fcc and hcp solid solutions in the al–tm (tm= ti, zr and hf) systems: a comparison of cluster expansion and supercell methods, Acta Materialia 56 (13) (2008) 3202–3221.

[23] A. van de Walle, The alloy-theoretic automated toolkit (atat): A user guide, Brown University, Providence, RI, Tech. Rep (2019).

[24] R. Singh, A. Sharma, P. Singh, G. Balasubramanian, D. D. Johnson, Accelerating computational modeling and design of high-entropy alloys, Nature Computational Science 1 (1) (2021) 54–61.

[25] S. Yin, J. Ding, M. Asta, R. O. Ritchie, Ab initio modeling of the energy landscape for screw dislocations in body-centered cubic high-entropy alloys,

npj Computational Materials 6 (1) (2020) 110.
[26] J. Cohen, M. Fine, Some aspects of short-range order, J. Phys. Radium 23 (10) (1962) 749–762.
[27] S. Zhao, Local ordering tendency in body-centered cubic (bcc) multi-principal element alloys, Journal of Phase Equilibria and Diffusion 42 (5) (2021) 578–591.
[28] R. Salloom, M. I. Baskes, S. G. Srinivasan, Atomic level simulations of the phase stability and stacking fault energy of fecocrmnsi high entropy alloy, Modelling and Simulation in Materials Science and Engineering 30 (7) (2022) 075002.
[29] X. Liu, J. Zhang, Z. Pei, Machine learning for high-entropy alloys: Progress, challenges and opportunities, Progress in Materials Science 131 (2023) 101018.
[30] I. Toda-Caraballo, J. S. Wróbel, D. Nguyen-Manh, P. Pérez, P. Rivera-Díaz-del Castillo, Simulation and modeling in high entropy alloys, JOM 69 (2017) 2137–2149.
[31] J. Rickman, G. Balasubramanian, C. Marvel, H. Chan, M.-T. Burton, Machine learning strategies for high-entropy alloys, Journal of applied physics 128 (22) (2020).
[32] V. Sorkin, S. Chen, T. L. Tan, Z. Yu, M. Man, Y. Zhang, First-principles-based high-throughput computation for high entropy alloys with short range order, Journal of Alloys and Compounds 882 (2021) 160776.
[33] P. A. Santos-Florez, S.-C. Dai, Y. Yao, H. Yanxon, L. Li, Y.-J. Wang, Q. Zhu, X.-X. Yu, Short-range order and its impacts on the bcc monbtaw multi-principal element alloy by the machine-learning potential, Acta Materialia 255 (2023) 119041.
[34] J. B. Seol, W.-S. Ko, S. S. Sohn, M. Y. Na, H. J. Chang, Y.-U. Heo, J. G. Kim, H. Sung, Z. Li, E. Pereloma, et al., Mechanically derived short-range order and its impact on the multi-principal-element alloys, Nature communications 13 (1) (2022) 6766.
[35] Y. Wu, F. Zhang, X. Yuan, H. Huang, X. Wen, Y. Wang, M. Zhang, H. Wu, X. Liu, H. Wang, et al., Short-range ordering and its effects on mechanical properties of high-entropy alloys, Journal of Materials Science & Technology 62 (2021) 214–220.
[36] D. Liu, Q. Wang, J. Wang, X. Chen, P. Jiang, F. Yuan, Z. Cheng, E. Ma, X. Wu, Chemical short-range order in fe50mn30co10cr10 high-entropy alloy, Materials Today Nano 16 (2021) 100139.
[37] X.-S. Yang, S. Deb, Cuckoo search via lévy flights, in: 2009 World congress on nature & biologically inspired computing (NaBIC), IEEE, 2009, pp. 210–214.
[38] F. Zhang, S. Zhao, K. Jin, H. Xue, G. Velisa, H. Bei, R. Huang, J. Ko, D. Pagan, J. Neuefeind, et al., Local structure and short-range order in a nicocr solid solution alloy, Physical review letters 118 (20) (2017) 205501.
[39] J. Li, Q. Fang, B. Liu, Y. Liu, Y. Liu, Mechanical behaviors of alcrfecuni high-entropy alloys under uniaxial tension via molecular dynamics simulation, RSC advances 6 (80) (2016) 76409–76419.
[40] Y. Afkham, M. Bahramyan, R. T. Mousavian, D. Brabazon, Tensile properties of alcrcofecuni glassy alloys: A molecular dynamics simulation study, Materials Science and Engineering: A 698 (2017) 143–151.
[41] Y. Tang, D. Li, Nano-tribological behavior of high-entropy alloys crmnfeconi and crfeconi under different conditions: A molecular dynamics study, Wear 476 (2021) 203583.
[42] U. Dahlborg, J. Cornide, M. Calvo-Dahlborg, T. Hansen, A. Fitch, Z. Leong, S. Chambreland, R. Goodall, Structure of some cocrfeni and cocrfenipd multicomponent hea alloys by diffraction techniques, Journal of Alloys and Compounds 681 (2016) 330–341.