A Novel Metaheuristic Based on the Nuclear Chain Reaction Process*

A. Mateos[1] and I. Zamorano[1]

***Abstract*— The Nuclear Chain Reaction (NCR) metaheuristic is introduced as an innovative optimization technique inspired by the fundamental principles of nuclear reactions. It is specifically designed to tackle high-dimensional and complex optimization problems. This algorithm emulates key aspects of nuclear chain reactions, such as fission, neutron moderation, and criticality control, to efficiently guide the search for optimal solutions. By adaptively segmenting the search space based on the fitness of evaluated solutions, NCR enables a more precise and targeted exploration.**

The algorithm's effectiveness is rigorously assessed through extensive evaluations on 20 numerical benchmark functions and compared against nine well-established optimization algorithms. The results demonstrate superior performance in terms of solution quality across most benchmark functions, highlighting NCR's robustness and adaptability. Furthermore, its scalability and general applicability indicate significant potential for addressing real-world complex problems. This study provides a valuable contribution to the field of optimization, positioning NCR as a promising tool for future research and practical applications.

I. INTRODUCTION

Metaheuristic algorithms are high-level optimization techniques designed to efficiently explore large and complex search spaces, particularly in scenarios involving uncertainty, incomplete information, and computational constraints [19]. Unlike exact methods, which become impractical for high-dimensional problems, metaheuristics use intelligent sampling strategies to balance exploration and exploitation, enhancing search efficiency [18].

One of the key advantages of metaheuristics is their broad applicability across various domains, as they do not require specific mathematical properties such as convexity or differentiability. This makes them suitable for both combinatorial and continuous optimization problems, with successful applications in logistics, engineering, medicine, and finance. They have been particularly effective in solving NP-hard problems such as the Traveling Salesman Problem [11], the Vehicle Routing Problem [7], and Job Shop Scheduling [16], where exact methods struggle due to computational complexity.

Although metaheuristics do not guarantee global optimality, they employ stochastic search mechanisms to prevent premature convergence and enhance solution diversity. This allows them to efficiently approximate high-quality solutions efficiently. Recent research [12] has focused on their theoretical properties, including convergence analysis and performance guarantees, although they remain primarily heuristic, non-deterministic methods aimed at near-optimal solutions.

In practice, metaheuristics have been instrumental in optimizing radiation therapy planning, disease diagnosis, and medical imaging. In engineering, they are widely used for trajectory planning, fluid dynamics, and robotics. They also contribute to finance by improving portfolio optimization, risk management, and algorithmic trading strategies. Despite their power, metaheuristics are not universally superior to exact methods; the choice depends on problem requirements, computational feasibility, and accuracy needs. Over time, numerous metaheuristics have emerged, including Genetic Algorithms [4], Particle Swarm Optimization [8], and Simulated Annealing [5][6], each inspired by different natural processes. This study introduces the Nuclear Chain Reaction metaheuristic, a novel approach based on nuclear reaction principles. The following sections present its theoretical foundations, numerical evaluations, and a comparative performance analysis against existing optimization techniques, demonstrating improved performance in terms of solution quality.

II. THE NUCLEAR CHAIN REACTION (NCR) ALGORITHM

A. Fundamentals of Nuclear Fission and Chain Reactions

Nuclear fission [13] occurs when the nucleus of a heavy atom, such as *uranium-235*, splits into smaller nuclei, releasing a large amount of *energy* and *free neutrons*. This reaction is the basis of *nuclear energy* production and nuclear weapons. However, uranium-235 is rare, making up less than 1% of natural uranium, whereas *uranium-238*, the most abundant isotope, is *not fissile* with slow neutrons and requires *fast neutrons* to undergo fission. When a uranium-235 nucleus captures a *neutron*, it forms an *unstable uranium-236 nucleus*, which then *splits*, generating *barium-141* and *krypton-92* as *fission products*, along with additional neutrons that can induce further fission, triggering a *nuclear chain reaction*. This self-sustaining process depends on the number of neutrons produced and absorbed during each fission event, which is measured by the *neutron multiplication factor* (k). This factor determines the behavior of the chain reaction:

- *Subcritical* ($k < 1$): The reaction dies out as fewer neutrons are available to sustain it.
- *Critical* ($k = 1$): The reaction remains stable, essential for nuclear reactor operation.
- *Supercritical* ($k > 1$): The reaction accelerates uncontrollably, potentially leading to nuclear explosions.

*This paper was supported by the Spanish Ministry of Science and Innovation project: PID2021-122209OB-C31.

[1]Alfonso Mateos and Iago Zamorano are with the Departamento de Inteligencia Artificial, Universidad Politécnica de Madrid, Campus de Montegancedo S/N, Boadilla del Monte, 28660 Madrid, Spain alfonso.mateos@upm.es

In nuclear reactors, this process is regulated using control rods (made of neutron-absorbing materials like boron or cadmium) to adjust the reaction rate and ensure stable and safe operation.

B. Nuclear Chain Reaction Algorithm: Structural and Dynamic Mechanisms

Inspired by the controlled dynamics of nuclear fission, the Nuclear Chain Reaction (NCR) algorithm is a metaheuristic optimization method that models a regulated chain reaction to efficiently explore complex search spaces. Its key mechanisms include:

1) *Fission-Based Exploration*: High-quality solutions "split" to generate new candidate solutions, mimicking neutron propagation in a nuclear chain reaction. This mechanism prevents premature convergence and enhances search diversity.
2) *Neutron Moderation*: Adaptive perturbation strategies refine candidate solutions, balancing exploration and exploitation while avoiding excessive divergence.
3) *Criticality Control*: The search intensity is dynamically adjusted:
   - *Subcritical sate ($k < 1$)*: If the process stagnates, additional perturbations are introduced to increase diversity.
   - *Supercritical state*: If the search becomes overly aggressive, adjustments are made to stabilize convergence.

By balancing global exploration and local refinement, the NCR algorithm efficiently navigates high-dimensional and complex problem landscapes. Its self-regulating nature makes it highly well-suited for fields such as engineering, logistics, bioinformatics, and computational physics.

The NCR algorithm is population-based and inspired by the dynamics of nuclear fission. It maintains equilibrium between exploration (global search) and exploitation (local refinement), drawing analogies from the operation of nuclear reactors. The following sections describe the fundamental components of the algorithm, including its structure, dynamic adaptation, and mathematical formulation.

1) *Population-Based Structure*: Analogous to a *nuclear reactor core* containing a regulated assembly of fissile isotopes, the *NCR algorithm* maintains a *population of candidate solutions*. Each solution acts as a *nuclear isotope* that interacts dynamically to drive the optimization process. The *population size* (F) remains *constant* throughout execution to ensure *computational stability*. Each iteration represents a *reaction cycle*, during which solutions undergo *fission-inspired transformations*. These are guided by *fitness-dependent criteria* to enhance the search process. The specific encoding of solutions is not essential, as long as the fitness function yields a numeriical value to be minimized or maximized.
2) *Fission Mechanism and Exploration Dynamics*: *Exploration* is modeled as the splitting of high-fitness solutions, triggering an *optimization cascade*. High-performing solutions act as *fissile nuclei*, releasing a fitness-proportional number of *neutrons* (k), which represent *potential new solutions*. These neutrons can:
   - *Induce further fissions*, refining the search around promising areas..
   - *Generate entirely new solutions*, increasing diversity.

   This mechanism ensures that *strong candidates* guide the optimization while maintaining *randomness* to explore new regions effectively.
3) *Search Space Enrichment and Adaptive Exploration*: The search space is segmented into *regions of varying enrichment levels*, akin to fissile material distribution in nuclear reactors. This guides the search as follows:
   - *Highly enriched regions*: Contain many high-fitness solutions, encouraging *intensified exploitation*.
   - *Poorly enriched regions*: Contain fewer promising solutions, prompting broader exploration.

   The behavior of newly generated solutions (neutrons) depends on their origin's enrichment level, influencing their role in the optimization.This is quantified using the *enrichment rate*:

   $$r_n = \frac{f_n - f^-}{f^* - f^-}, \tag{1}$$

   where: f_n is the fitness of the solution n, f^* is the best fitness found so far, and f^- is the worst fitness in the population. This scaling ensures that solutions from *high-fitness regions* contribute significantly, while those from low-fitness areas undergo stronger perturbations or are discarded.
4) *Neutron Moderation and Scattering Dynamics*: In nuclear physics, *moderation* slows for neutrons to increase the likelihood of further fission. In NCR, a *moderation factor* α regulates the trade-off between *local* and *global search*. The *scattering probability* p_n of a new solution is defined as:

   $$p_n = 1 - \alpha * r_n = 1 - \alpha * \frac{f_n - f^-}{f^* - f^-}, \tag{2}$$

   The parameteri α determines the exploration degree:
   - *Lower* α : Favors global exploration by enabling large deviations.
   - *Higher* α: Emphasizes local refinement with small perturbations.

   This mechanism allows *dynamic regulation of of search intensity* according to the problem landscape.
5) *Generation of New Solutions via Controlled Perturbation*: New solutions are generated using *probabilistic perturbation* influenced by the scattering probability. Each solution dimension can:
   - *Remain unchanged* (No variation),
   - *Undergo local perturbation* within neighborhood,

- *Receive a global perturbation* from the entire search space.

This is governed by the *Metropolis algorithm*, ensuring a balance between local refinement (close impacts) and global exploration (*distant impacts*):

- *Close impacts*: Small adjustments in high-fitness regions allow fine-tuned optimization while maintaining stability. *Algorithm 1* illustrates this behavior, showing how the conditions for making a change in each dimension x_i of the solution are evaluated:

Algorithm 1 closeImpact

Require: $(x_1, \ldots, x_n), N(X)$, $0 > p_i > 1$
for all $x_i \in x$ **do**
  if $random\,([0, 1)) < p_i$ **then**
    $x'_i \leftarrow x_i$
  else
    $x'_i \leftarrow random\,[N(x_i)_{min}, N(x_i)_{max}\,]$
  end if
end for
return x'

Ihe parameter p_i is influenced by the solution's enrichment level. If the random value exceeds p_i, the variable is perturbed within the *local neighborhood*, maintaining a *balance between refinement* and *variation*.

- *Distant impacts.* Larger jumps across the search space prevent stagnation and enhance diversity. *Algorithm 2* describes this process for distant impacts, also using the *Metropolis algorithm* to evaluate each dimension of the solution and decide whether a broader variation is made:

Algorithm 2 distantImpact

Require: $(x_1, \ldots, x_n), N(X)$, $0 > p_i > 1$
for all $x_i \in x$ **do**
  if $random\,([0, 1)) < p_i$ **then**
    $x'_i \leftarrow random\,[N(x_i)_{min}, N(x_i)_{max}\,]$
  else
    $x'_i \leftarrow random\,[(x_i)_{min}, (x_i)_{max}\,]$
  end if
end for
return x'

Close impacts promote intensification by fine-tuning in promising areas, while *distant impacts* enhance diversity and prevent stagnation. The *scattering probability* p_i regulates this balance.

6) *Criticality Control and Selection Mechanism*: Like nuclear reactors, NCR incorporates control mechanisms to maintain population stability. The *effective neutron multiplication factor* k dictates whether the process is *subcritical*, *critical* or *supercritical*. Since $k > 1$, the population would grow exponentially.
To prevent uncontrolled growth, a *fitness-based roulette wheel selection* is used. Solutions are probabilistically selected for the next iteration based on fitness. An *elitism parameter* e ensures top-performing solutions are preserved.

7) *Stopping Criteria and Convergence Conditions*: The algorithm terminates when one or more stopping conditions are met:
   - Reaching a maximum number of iterations,
   - Achieving a satisfactory fitness threshold,
   - Detecting stagnation (minimal improvement over several iterations).

The NCR algorithm presents a physic-inspired, self-adaptive optimization framework that dynamically balances *exploration and exploitation* using *nuclear reaction principles*. Its population-based structure, combined with *fitness-dependent adaptation*, *self-regulating search intensity*, and *criticality-controlled selection*, enables efficient navigation of complex optimization landscapes.

C. Step-by-Step Procedure of the NCR Algorithm

The step-by-step execution of the NCR algorithm is rigorously detailed in *Algorithm 3*. Both representations systematically describe the iterative process of solution evolution, from initialization to final selection. This detailed description ensures transparency and reproducibility.

Algorithm 3 Nuclear chain reaction algorithm

Require: $F > 0$, $k > 1$, $0 > \alpha > 1$, $0 > e > 1$
$nuclei \leftarrow$ Set of size F of randomly initialized solutions
$sol^* \leftarrow$ solution with best fitness in $nuclei$; $f^* \leftarrow fitness(sol^*)$
$sol^- \leftarrow$ solution with worst fitness in $nuclei$; $f^- \leftarrow fitness(sol^-)$
$E \leftarrow \lfloor F * e \rfloor$
while not meeting stopping criteria **do**
  $impacts = \emptyset$
  for all $n \in nuclei$ **do**
    $f_n \leftarrow fitness(n)$
    if $f_n > f^*$ **then**
      $sol^* \leftarrow n; f^* \leftarrow f_n$
    else if $f_n < f^-$ **then**
      $sol^- \leftarrow n; f^- \leftarrow f_n$
    end if
    $p_n \leftarrow 1 - \left(\alpha * \frac{f_n - f^-}{f^* - f^-}\right)$
    for $k' = 1, \ldots, k$ **do**
      if $random(0, 1) < p_n$ **then**
        $impacts \leftarrow impacts \bigcup \{closeImpact(n, p_n)\}$
      else
        $impacts \leftarrow impacts \bigcup \{distantImpact(n, p_n)\}$
      end if
    end for
  end for
  $impacts \leftarrow sort(impacts, fitness)$
  $nuclei \leftarrow impacts[1, \ldots, E] \bigcup rouletteSelection(impacts[E + 1, \ldots, kF], fitness, F - E)$
end while
return f^*, sol^*

Algorithm Input Parameters (Requirements)

The NCR algorithm is controlled by key parameters that manage population dynamics, solution variation, and selection:

- $F > 0$: Fixed population size, representing candidate solutions (nuclei).
- $k > 1$: Number of neutrons per solution (multiplier for solution copies).
- $0 < \alpha < 1$: Sensitivity factor, regulates probabilistic selection.
- $0 < e < 1$: Elitism rate, controls number of top-performing solutions retained.

Step-by-Step Execution of the NCR Algorithm 3

1) *Initialization* (Nuclei Formation): A population of F randomly generated candidate solutions (nuclei) is created (line 1).
2) *Identifying Best and Worst Solutions*: The algorithm identifies the best-performing (sol^{*}) and worst-performing (sol^{-}) solutions to normalize fitness values (lines 2 and 3).
3) *Elitism Threshold Calculation*: A fraction $E = \lfloor F \times e \rfloor$ of the top solutions is preserved for the next generation (line 4).
4) *Main Iteration Loop*: The algorithm iterates until a stopping criterion (e.g., convergence or max iterations) is met (line 5).
5) *Impact List Reset*: A temporary list stores newly generated solutions in each iteration (line 6).
6) *Fitness Evaluation*: Each candidate solution is evaluated to determine its fitness value (line 7).
7) *Best and Worst Solutions Update*:
   - If a new solution surpasses sol^{*}, it replaces it (lines 9–10).
   - If a new solution is worse than sol^{-}, it replaces it(lines 11–12).
8) *Scattering Probability Calculation*: The probability p_n determines whether new solutions undergo local or global variation, adapting based on fitness differences (line 14).
9) *Generation of New Solutions*:
   - Each solution undergoes fission-like reproduction, generating k new candidates.
   - The variation type is determined by p_n:
     – If $r < p_n$, a local variation is applied (Algorithm 1) (lines 16–17).
     – Otherwise, a global variation is applied (Algorithm 2) (lines 18–19).
   - The newly generated solutions are stored in the *impacts* list.
10) *Sorting by Fitness*: The new and existing solutions are ranked in descending order of fitness (line 23).
11) *Selection of the Next Generation*:
    - The top E elite solutions are preserved.
    - The remaining positions are filled using roulette wheel selection, favoring higher-fitness solutions (line 24).
12) *Loop Continuation*: The process repeats until the stopping condition is met. (lines 5–25).
13) *Final Output*: The algorithm returns the best solution sol^{*} and its fitness value. f^{*} (line 26).

III. Benchmark Functions Tests

In this section, we present the results of numerical experiments conducted to assess the performance of the proposed NCR algorithm across a set of standard benchmark functions. These tests aim to validate NCR's effectiveness in solving both low- and high-dimensional optimization problems and compare its performance with existing metaheuristics.

Benchmark Functions

A total of 20 numerical benchmark functions were selected for the experiments. These functions, originally utilized by Azizi in his comparison of the atomic orbital search (AOS) metaheuristic in 2021 [1], provide a comprehensive evaluation platform. The functions are: Becker-Lago function (f_1), Carrom table function (f_2), Cube function (f_3), Himmelblau function (f_4), Levy 5 function (f_5), Ursem 3 function (f_6), Ursem 4 function (f_7), Ackley 1 function (f_8), Alpine 1 function (f_9), Deb 3 function (f_{10}), Exponential function (f_{11}), Inverted cosine wave function (f_{12}), Schwefel 2.20 function (f_{13}), Stretched V Sine Wave function (f_{14}), Deb 1 function (f_{15}), Levy 8 function (f_{16}), Mishra 11 function (f_{17}), Salomon function (f_{18}), Schwefel 2.21 function (f_{19}) and Stepint function (f_{20}).

Each of these functions presents distinct characteristics, making them ideal for testing an algorithm's ability to handle multimodal landscapes, deceptive optima, separability, and dimensionality scalabitity.

Metaheuristic Comparisons

To contextualize the NCR algorithm's performance, we compared it with le following well-established metaheuristics: Genetic Algorithm (GA) [4], Multi-Verse Optimization (MVO) [10], Sine Cosine Algorithm (SCA) [9], Bat-Inspired Algorithm (BIA) [14], the Big-Bang Big-Crunch (BBBC) algorithm [3], the Cuckoo Search optimization Algorithm (CSA) [15], the Wind-Driven Optimization (WDO) algorithm [2], Atom Search Optimization (ASO) [17], and Atomic Orbital Search (AOS) [1].

The parameters for these algorithms were sourced from the respective literature [1], while the NCR algorithm's specific parameters for each benchmark function were fine-tuned to optimize the algorithm's performance across different test functions.

Each function was evaluated over 100 independent runs, with a maximum of 150000 objective function evaluations. The statistical results are shown in Table I, which highlights the difference between the best solution obtained from the 100 evaluations and the known optimal solution for each benchmark function.

TABLE I

DIFFERENCE BETWEEN THE BEST SOLUTION AND THE OPTIMUM

N_o	GA	MVO	SCA	BIA	BBBC	CSA	ASO	WDO	AOS	NCR	Avg.
f_1	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
f_2	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
f_3	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
f_4	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
f_5	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
f_6	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
f_7	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
f_8	0.2	0.2	0.1	14	13.5	3.3	3.4	**0**	**0**	0.4	3.5
f_9	**0**	1.9	2	3.1	2.5	9.2	9.8	**0**	**0**	**0**	2.9
f_{10}	**0**	**0**	0.4	0.4	**0**	0.1	0.2	0.3	0.1	**0**	0.2
f_{11}	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
f_{12}	16	19	35	30	11.3	7.3	34	15	0.1	**0**	16.9
f_{13}	2.2	0.2	2.1	465	462	2.3	0.3	**0**	**0**	**0**	93.5
f_{14}	5.4	1.3	2.8	**0.01**	2.6	7.7	17	1.7	0.3	4.8	4.4
f_{15}	**0**	**0**	0.5	0.5	**0**	0.2	0.1	0.03	0.1	**0**	0.1
f_{16}	20	37	46	6.1	58.5	75	5.9	6.3	2.9	0.2	25.7
f_{17}	**0**	**0**	0.2	0.2	**0**	**0**	**0**	**0**	**0**	**0**	0.04
f_{18}	2.1	0.8	**0**	17	17	2.9	4	0.1	**0**	0.3	4.4
f_{19}	18	8.8	58	32	27.5	4.9	12	0.1	**0**	0.1	16
f_{20}	**0**	63	218	281	11	14	216	312	229	**0**	144

Key Insights from the Experiments:

1) *High Precision*:
   - NCR consistently reaches or closely approaches the global optimum, even in complex high-dimensional cases (e.g., f_8, f_{16}, f_{18}, f_{19}).
   - It adapts well to varying problem complexities and high-dimensional spaces.
   - It outperforms the genetic algorithm, the sine cosine algorithm, and atom search optimization in most cases, significantly reducing the gap to the optimum.
2) *Superior Performance*: NCR achieves the optimal solution in 15 out of 20 benchmark functions, outperforming atomic orbital search (14 optima), genetic algorithm (13 optima), and multi-verse optimization, big-bang big-crunch, and wind-driven optimization (11 each).
3) *Consistent Outperformance:* NCR surpasses cuckoo search optimization, atom search optimization, and the bat-inspired algorithm in all functions except f_{14} (where the bat-inspired algorithm performs better). Furthermore, It also outperforms the genetic algorithm, multi-verse optimization, big-bang big-crunch, and the sine cosine algorithm in all but f_8, f_{14}, and f_{18} (in which the sine cosine algorithm shows superior performance).
4) *Best Performance on* f_{20} : NCR and the genetic algorithm both reach the optimum for f_{20} while other metaheuristics remain significantly farther from the best solution.

The study also compares the *average performance* of the algorithms over 100 evaluations per benchmark function, focusing on the difference between the obtained solutions and their respective optima. Unlike best-case results, these averages provide a more realistic assessment of each metaheuristic's typical performance. A more detailed analysis follows:.

1) *Convergence in 2D Functions*: NCR, cuckoo search optimization, and atom search consistently achieve full convergence in the first seven 2D functions, reaching the optimal solution in every trial.
   In function f_{11}, all 10 metaheuristics converge to the optimum, indicating it's relatively easy for these algorithms to solve.
2) *Performance in Higher Dimensions*: As dimensionality increases, convergence becomes more difficult. However, NCR either reaches or gets very close to the optimum in most complex functions. It outperforms many other algorithms in functions like f_8, f_{12}, f_{14}, f_{16}, f_{18} and f_{19}.
3) *NCR's Competitive Edge*: NCR ties with atomic orbital search, reaching the optimum in 14 out of 20 functions. It outperforms other algorithms such as the genetic algorithm, multi-verse optimization, cuckoo search optimization, and wind-driven optimization, which reach the optimum in only 9 functions.
4) *Outperforming Other Metaheuristics*: NCR consistently outperforms the genetic algorithm, the sine cosine algorithm, and wind-driven optimization across all benchmark functions, demonstrating its competitiveness against both well-established and newer algorithms.
5) *Functions Where NCR is Surpassed*: NCR is outperformed by specific algorithms in a few functions:
   - f_{14}: Surpassed by the multi-verse optimization, the bat-inspired algorithm, and big-bang big-crunch.
   - f_{12}: Cuckoo search optimization performs better.
   - f_8, f_{14} and f_{18}: Wind-driven optimization and atomic orbital search surpass NCR, though it remains highly competitive.
6) *Outstanding Performance for Specific functions*: NCR excels in certain functions:
   - f_{13}: Reaches the optimum alongside atomic orbital search and wind-driven optimization.
   - f_{15}: Fully converges to the optimum, matching the big-bang big-crunch algorithm.
   - f_{20}: Reaches the optimum with the genetic algorithm, outperforming all other metaheuristics. NCR demonstrates robustness in high-dimensional and complex landscapes.

IV. CONCLUSIONS

This study provides a comprehensive analysis of the *Nuclear Chain Reaction (NCR) metaheuristic*, examining its formulation, performance characteristics, and applicability across a wide spectrum of optimization problems. The empirical findings demonstrate that NCR consistently outperforms several well-established metaheuristic algorithms when evaluated on diverse benchmark functions. Its capacity to effectively address complex optimization challenges with high reliability and computational efficiency underscores its potential as a powerful optimization framework, particularly in scenarios where precision, robustness, and convergence stability are paramount. A key attribute of NCR is its *remarkable versatility*, enabling it to perform effectively across a broad range of optimization landscapes, from low-dimensional search spaces to highly complex, high-dimensional problem instances. The algorithm exhibits strong global search capabilities, consistently converging towards high-quality solutions with notable accuracy, even in scenarios where other algorithms are prone to premature convergence. Furthermore, NCR's low standard deviation across multiple experimental runs highlights its robustness, ensuring consistent and repeatable performance - an especially valuable trait in real-world applications where *solution stability* is critical.

One of the fundamental drivers of NCR's success is its *well-balanced trade-off between exploration and exploitation*. The algorithm strategically integrates *region enrichment mechanisms*, allowing it to efficiently explore the solution space while mitigating the risk of becoming trapped in local optima. This balance is carefully regulated through key algorithmic parameters, including the *population size* (F), *neutron multiplication factor* (k), *moderation factor* (α), and *elitism rate* (e). The fine-tuning of these parameters plays a pivotal role in ensuring a *diverse yet focused* search process, reinforcing NCR's adaptability across various optimization domains.

Despite its strong empirical performance, *further refinements* could enhance NCR's computational efficiency and extend its applicability. One notable challenge is the *computational cost* associated with solution evaluation and selection processes. Optimizations such as *partial sorting techniques* for elite preservation and more *computationally efficient selection mechanisms* could reduce algorithmic complexity without compromising solution quality. Additionally, expanding NCR's scope to *combinatorial optimization* and *multi-objective problem solving* would significantly broaden its utility in addressing real-world problems involving multiple conflicting objectives and constraints.

These prospective improvements open promising avenues for future research, reinforcing NCR's position as a *versatile and scalable optimization tool*. By further enhancing its efficiency and extending its applicability to more complex problem domains, NCR has the potential to become a *foundational algorithm in the field of global optimization*. Given its demonstrated accuracy, robustness, and adaptability, *NCR stands as a highly competitive framework for solving both theoretical and practical optimization challenges across diverse disciplines*.

REFERENCES

[1] M. Azizi, Atomic orbital search: A novel metaheuristic algorithm. Appl. Math. Modell., vol. 93, pp. 657-683, 2021.

[2] Z. Bayraktar, M. Komurcu, and D. H. Werner, Wind Driven Optimization (WDO): A novel nature-inspired optimization algorithm and its application to electromagnetics, In *IEEE Antennas and Propagation Society International symposium*, Toronto, ON, Canada, 2010, pp. 1-4.

[3] O.K. Erol, and I. Eksin, A new optimization method: big bang–big crunch. Adv. Eng. Softw., vol. 37(2), pp. 106-111, 2006.

[4] J.H. Holand, *Adaptation in Natural and Artificial Systems*. Ann Arbor: The University of Michigan Press, 32, 1975.

[5] S. Kirkpatrick, Jr, C. D. Gelatt & M. P. Vecchi, Optimization by simulated annealing. science, vol. 220(4598), pp. 671-680, 1983.

[6] V. Černý, Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm. *J. Optim. Theory Appl.*, vol. 45, pp. 41-51, 1985.

[7] N. Jozefowiez, F. Semet, E.G. Talbi, Multi-objective vehicle routing problems, Eur J. Oper. Res., vol. 189(2), pp. 293-309, 2008.

[8] J. Kennedy, and R. Eberhart, Particle swarm optimization. In Proceedings of ICNN'95-international conference on neural networks (Vol. 4, pp. 1942-1948). ieee, 1995.

[9] S. Mirjalili, SCA: a sine cosine algorithm for solving optimization problems, *Knowl.-Based Syst.* vol. 96, pp. 120–133, 2016.

[10] S. Mirjalili, S.M. Mirjalili, and A. Hatamlou, Multi-verse optimizer: a nature-inspired algorithm for global optimization. *Neural Comput. Appl.*, vol. 27, pp. 495-513, 2016.

[11] K. Panwar, and K. Deep, Discrete Grey Wolf Optimizer for symmetric travelling salesman problem. *Appl. Soft Comput.*, vol. 105, pp. 107298, 2021.

[12] K. Rajwar, K. Deep, and S. Das, An exhaustive review of the metaheuristic algorithms for search and optimization: Taxonomy, applications, and open challenges. *Artif. Intell. Rev.*, vol. 56(11), pp. 13187-13257, 2023.

[13] R.A. Serway and J.W. Jewett, Physics for Scientists and Engineers, Cengage Learning, 2018.

[14] X.S. Yang, A new metaheuristic bat-inspired algorithm, in *Nature Inspired Cooperative Strategies for =ptimization (NICSO 2010)*, pp. 65-74. Berlin, Heidelberg: Springer, 2010, Berlin Heidelberg.

[15] X.S. Yang, and S. Deb, Cuckoo search via Lévy flights, in *2009 World Congress on Nature & Biologically Inspired Computing (NaBIC)*, Coimbatore, India, 2009, pp. 210-214.

[16] H. Zhang, S. Liu, S. Moraca, and R. Ojstersek, An effective use of hybrid metaheuristics algorithm for job shop scheduling problem. *Int. J. Simul. Process Model.*, vol. 16(4), pp. 644-657, 2017.

[17] W. Zhao, L. Wang, and Z. Zhang, A novel atom search optimization for dispersion coefficient estimation in groundwater. *Future Gener. Comput. Syst.*, vol. 91, pp. 601-610, 2019.

[18] F. Peres and M. Castelli, Combinatorial optimization problems and metaheuristics: Review, challenges, design, and development. *Appl. Sci.*, vo. 11(14), pp. 6449, 2021.

[19] K. Sörensen and F. Glover, Metaheuristics. Encycl. Oper. Res. Manag. Scii, vol. 62, pp. 960-970, 2013.