An Immune Memetic Algorithm for Dynamic Optimization Problems (Postprint)

Authors: Yang Zhou, Yuan Yichuan, Luo Tingxing, Qin Jin

Date: 2018-05-24T00:00:00+00:00

Abstract

To address the deficiencies of traditional immune network dynamic optimization algorithms, including weak local search capability, low optimization accuracy, and proneness to premature convergence, this paper proposes an immune memetic algorithm for solving dynamic optimization problems. Based on the fundamental framework of memetic algorithms, the artificial immune network algorithm is utilized as the global search mechanism, while the tabu search algorithm serves as the local search operator; furthermore, Cauchy mutation is incorporated to strengthen the algorithm's global search ability and effectively mitigate premature convergence. Experimental evaluations on a benchmark suite of classic dynamic optimization functions under identical conditions indicate that the proposed immune memetic algorithm achieves superior search precision and convergence speed compared to other analogous algorithms.

Full Text

An Immune Memetic Algorithm for Dynamic Optimization Problems

YANG Zhou[1], YUAN Yichuan[1], LUO Tingxing[2], QIN Jin[1]
(1. College of Computer Science & Technology, Guizhou University, Guiyang 550025, China;
2. Guiyang Information Industry Development Center, Guiyang 550081, China)

Abstract: Traditional immune network dynamic optimization algorithms suffer from weak local search capability, low optimization precision, and premature convergence. To address these limitations, this paper proposes an artificial-immune-network-based memetic algorithm for dynamic optimization problems. Built upon the memetic algorithm framework, the proposed method employs an artificial immune network algorithm for global search and integrates a tabu search algorithm as the local search operator. Additionally, Cauchy mutation is introduced to enhance global search capability and effectively prevent premature

convergence. Experimental results on classic dynamic optimization benchmark functions under identical conditions demonstrate that the proposed immune memetic algorithm achieves superior search precision and convergence speed compared to other similar algorithms.

Keywords: dynamic optimization; artificial immune; tabu search; Cauchy mutation

0 Introduction

Many real-world optimization problems exhibit dynamic characteristics. For instance, in scheduling problems, new tasks continuously arrive, raw material quality varies over time, machines gradually wear out, and unexpected failures may occur at any moment. Dynamic optimization problems (DOPs) can be defined as follows: let X represent the search space, t represent time, and $f(x,t)$ represent the objective function. $S(t)$ denotes the feasible solution set at time t, and $f(x,t)$ represents the objective function value of candidate solution x at time t. In other words, certain objectives or constraints in DOPs change continuously over time—a poor choice at one moment might become favorable later, while previously insignificant conditions may suddenly gain importance. As society evolves rapidly, DOPs have expanded to encompass nearly all aspects of engineering, science, economy, and daily life, attracting increasing attention from researchers.

The key distinction between dynamic and static optimization lies in that static problems only require convergence to a global optimum, whereas DOPs demand algorithms that not only locate the global optimum at each time instance but also rapidly track the changing optimal solution. This necessitates that dynamic optimization algorithms possess both fast convergence and diversity maintenance capabilities. Currently, the main approaches for solving DOPs fall into two categories: evolutionary algorithms and swarm intelligence algorithms. Evolutionary algorithms, inspired by biological evolution mechanisms, are traditionally used for static optimization. Researchers have adapted them for DOPs by incorporating strategies such as diversity preservation, memory mechanisms, prediction schemes, and multi-population approaches. Swarm intelligence algorithms mimic the behavior and interactions of biological groups, including ant colony optimization, particle swarm optimization, firefly algorithm, and fish swarm algorithm. Recent developments have spawned novel heuristic algorithms, such as the fireworks algorithm inspired by firework explosions.

Artificial immune systems, which emulate the biological immune system's functions, have been widely applied in optimization, fault diagnosis, and control. The memory mechanisms, clonal variation, and network interactions characteristic of artificial immune algorithms often yield superior optimization performance compared to traditional methods. The artificial immune algorithm framework offers extensive room for improvement through mechanism integra-

tion and hybridization with other algorithms. For example, Liu et al. proposed a Cooperative Artificial Immune Network (CoAIN) for multimodal function optimization, which enhances search capability by incorporating collaborative mechanisms from particle swarm optimization, enabling individuals in the network to cooperate rather than merely suppress each other. Zhao Hui et al. developed a distributed elite evolutionary model-based artificial bee colony immune algorithm that combines bee colony concepts with immune clonal selection to overcome the slow convergence and low search precision of conventional artificial bee colony algorithms.

Memetic algorithms (MA) have emerged as a hot research topic in evolutionary computation in recent years. By integrating global search with local search, MA provides an effective framework for balancing exploration and exploitation in dynamic optimization. Building upon existing research, this paper proposes an immune memetic algorithm for dynamic optimization that addresses the low precision, slow convergence, and premature convergence issues observed in traditional artificial immune network algorithms for DOPs.

---

1 Related Research

1.1 The dopt-aiNet Algorithm

The artificial immune network algorithm, developed based on information processing mechanisms of biological immune systems, primarily involves immunological principles such as clonal selection, cell mutation, and immune cell networks. Initially applied to information compression and data clustering, Casto et al. first adapted the artificial immune network for optimization problems in 2002, proposing the opt-aiNet algorithm. Subsequently, they improved upon opt-aiNet to create the dynamic optimization artificial immune network algorithm (dopt-aiNet), which demonstrated promising performance in the 2009 IEEE CEC dynamic optimization competition.

The dopt-aiNet algorithm mimics the biological immune process where immune cells detect foreign antigens, identify them, stimulate clonal expansion, and generate antibodies to eliminate antigens. The algorithmic flow proceeds as follows:
a) Initialize the population by generating an initial cell population and calculating individual fitness values.
b) Perform cell cloning by conducting clonal expansion for each cell individual.
c) Apply cell mutation through Gaussian mutation on cloned cells.
d) Conduct elite selection by retaining the cell with the highest fitness value in each mutated group.
e) Perform suppression judgment: if the cell population meets suppression conditions, proceed; otherwise, go to step h.
f) Execute cell suppression: between two cells with suppression distance below the threshold, eliminate the individual with lower fitness.
g) Introduce new cells: randomly generate new cell individuals proportionally

and add them to the current population.
h) Termination condition check: if termination criteria are met, output results; otherwise, return to step b.

1.2 Tabu Search Algorithm

Tabu search (TS) is a metaheuristic search algorithm that mimics human memory function by using a tabu list to block recently searched regions, thereby avoiding cyclic search and enabling escape from local optima. Compared to traditional optimization algorithms, TS exhibits stronger "hill-climbing" capability and local search ability. TS has been widely applied to combinatorial optimization problems such as the traveling salesman problem and 0-1 knapsack problem, with recent efforts extending its application to function optimization.

TS is essentially an excellent neighborhood search algorithm with fast convergence speed. However, its convergence rate and solution quality heavily depend on the initial solution quality—a good initial solution can facilitate rapid convergence to the global optimum, while a poor one may significantly degrade convergence speed. The key components of TS include neighborhood definition, tabu list design, and aspiration criteria. The neighborhood embodies the local search philosophy, the tabu list prevents cyclic search, and the aspiration criteria provide feedback for promising solutions. The basic TS procedure involves searching the neighborhood of the current solution. If a better solution is found in the neighborhood, it replaces the current solution, and the search continues in the new neighborhood until termination. The TS algorithm flow can be summarized as:
a) Initialize parameters: provide initial solution X_0, calculate objective function value $f(X_0)$, and assign X_0 to current solution X.
b) Generate neighborhood and candidate set: create neighborhood $N(X)$ of current solution X, select a certain number of solutions as candidate set Y, and calculate their objective function values.
c) Check neighborhood search condition: if candidate set Y is not empty and optimal solution Y^* has not been selected, continue; otherwise, proceed to step e.
d) Apply tabu criterion: find the optimal solution Y^* from candidate set Y. If Y^* is not in the tabu list, assign it to current optimal solution; otherwise, remove it from the neighborhood and return to step c.
e) Update tabu list: if optimal solution Y^* is found, insert current solution X into the tabu list and assign Y^* to X.
f) Termination check: if termination condition is satisfied, stop; otherwise, return to step b.

2 Immune Memetic Algorithm for Dynamic Optimization

Memetic algorithms represent a hybrid framework based on Darwinian natural evolution theory and Dawkins' cultural evolution concepts. By combining evolutionary algorithms with local search methods, MA employs evolutionary algorithms for global search while using local search strategies to improve solution precision for certain individuals, thereby maintaining a balance between exploration and exploitation.

2.1 Algorithm Concept

The dopt-aiNet algorithm for dynamic optimization problems primarily employs operations such as initialization, clonal expansion, high-frequency mutation, and elite selection to optimize candidate solutions and ultimately find problem solutions, while maintaining population diversity through suppression mechanisms and introducing new individuals after suppression. Although a widely distributed population across the solution space often indicates strong global search capability, the algorithm tends to exhibit insufficient optimization capability and premature convergence when reaching certain optimization stages, affecting solution precision.

To address these limitations of dopt-aiNet in solving DOPs, this paper proposes an Artificial Immune Network-based Memetic Algorithm (AINMA) based on the memetic algorithm framework. AINMA leverages the local search advantages of tabu search and combines them with the strong global search capability of the artificial immune network algorithm. By introducing tabu search after the cell suppression phase, the artificial immune network provides good initial solutions for tabu search, thereby improving algorithmic search precision. Additionally, the algorithm integrates Cauchy mutation operator with traditional Gaussian mutation operator, switching between them during different optimization stages. Cauchy mutation is used in early stages to enable broader search of the solution space and prevent individuals from getting trapped in local optima, while Gaussian mutation is employed in later stages to prevent excessive mutation that might drive individuals away from the global optimum. The improved algorithm also employs detector cells to monitor environmental changes: after each iteration, the detector population calculates fitness values, and any change indicates environmental shift, prompting the optimization population to restart searching for new optima.

2.2 Cauchy Mutation Operator

Traditional Gaussian mutation operates on individual $X_i = (x_{i1}, x_{i2}, ..., x_{in})$ according to equation (2):

$$x'_{ij} = x_{ij} + \eta \cdot G(0,1)$$

where η is a constant controlling mutation step size, and $G(0,1)$ is a random

number generated from a Gaussian distribution with mean 0 and standard deviation 1.

The Cauchy mutation operator operates on individual X_i according to equation (3):

$$x'_{ij} = x_{ij} + \eta \cdot C(0,1)$$

where η is a constant controlling mutation step size, and $C(0,1)$ is a random number generated from a Cauchy distribution function.

Comparing the Gaussian and Cauchy density functions reveals that the Cauchy density function has heavier tails, meaning Cauchy mutation produces offspring that differ more significantly from their parents, giving individuals a higher probability of escaping local optima. Consequently, Cauchy mutation generally provides better global search capability than Gaussian mutation. This paper combines both mutation strategies: using Cauchy mutation in early optimization stages prevents premature stagnation, while switching to Gaussian mutation after suppression conditions are met (indicating the search has progressed sufficiently close to the global optimum) prevents excessive deviation from the global optimum.

2.3 Tabu Table, Neighborhood, and Aspiration Criteria

This work integrates tabu search into the dopt-aiNet algorithm. The immune algorithm's cloning, mutation, and selection operations provide good initial solutions for tabu search, which then serves as a local search tool to improve overall optimization precision. The following sections briefly introduce the tabu table, neighborhood, and aspiration criteria in the AINMA algorithm.

The tabu table is the core of tabu search. In AINMA, it stores recently visited neighborhood candidate solutions to prevent cyclic search and avoid getting trapped in local optima. The aspiration criterion prevents the algorithm from missing excellent solutions. In AINMA, when a neighborhood candidate solution's fitness surpasses the historical best fitness, the solution is accepted regardless of its tabu status.

Neighborhood partitioning is crucial in tabu search. Unlike combinatorial optimization problems with discrete solution spaces, function optimization problems feature continuous solution spaces. Therefore, this paper adopts the neighborhood partitioning method described in the literature [] for selecting neighborhood solutions.

For dynamic optimization problems, each static optimization subproblem is represented by the model in equation (4):

$$\begin{aligned} \min \quad & f(X) \\ \text{s.t.} \quad & L \leq X \leq U \end{aligned}$$

where $X = (x_1, x_2, ..., x_n)$ is an n-dimensional variable with lower bounds $L = (l_1, l_2, ..., l_n)$ and upper bounds $U = (u_1, u_2, ..., u_n)$. Concentric hyper-rectangles divide each component x_j of the current solution into k spaces, with each space partitioned according to equation (5):

$$R_{ij} = \{x'_{ij} \mid l_j + (r_i - 1) \cdot \frac{u_j - l_j}{k} < x'_{ij} < l_j + r_i \cdot \frac{u_j - l_j}{k}\}$$

for $i = 1, 2, ..., k$ and $j = 1, 2, ..., n$, where $r_i \in \{1, 2, ..., k\}$. Within each concentric rectangle, a point is randomly selected, and the k selected points form the neighborhood candidate solutions for current solution component x_j.

2.4 Algorithm Flow

The AINMA algorithm consists of the following steps:
a) Initialize population: generate optimization population and detector population, and set corresponding parameters.
b) Clone: perform clonal expansion on optimization population individuals.
c) Mutate: apply high-frequency mutation to each clone. If mutation flag is 0, use Cauchy mutation; otherwise, use Gaussian mutation, and calculate mutated individuals' fitness values.
d) Select: retain individuals with higher fitness in each mutated group.
e) Suppression check: determine if suppression condition is met; if so, proceed; otherwise, go to step i.
f) Suppress: set mutation flag to 1 and suppress the lower-fitness individual between two cells with distance below threshold.
g) Tabu search: generate neighborhood and candidate solution set for current best solution, find optimal solution in candidate set, and continuously update tabu table until termination condition is met.
h) Introduce random individuals: randomly generate new individuals and add them to current optimization population.
i) Detect environmental change: use detector population to monitor environmental changes. If change occurs, set mutation flag to 0; otherwise, continue.
j) Termination check: if termination condition is met, output results; otherwise, return to step b.

The AINMA algorithm flowchart is shown in Figure 1 [Figure 1: see original paper].

---

3 Experimental Results and Analysis

3.1 Experimental Setup

To validate the performance of the proposed immune memetic dynamic optimization algorithm, we conducted simulation tests using the generalized dynamic

benchmark generator (GDBG) function test suite proposed in the literature []. The GDBG test suite comprises six function types, each with seven change types including small changes, large changes, random changes, chaotic changes, periodic changes, periodic changes with noise, and dimensional changes. This dynamic function test suite is constructed by rotating and combining the five base functions shown in Table 1 to form six problem classes of varying difficulty:

- F1: Rotation peak function (with 10 and 50 peaks)
- F2: Composition of Sphere's function
- F3: Composition of Rastrigin's function
- F4: Composition of Griewank's function
- F5: Composition of Ackley's function
- F6: Hybrid composition function

F1 is a maximization problem, while F2-F6 are minimization problems. Table 1 shows the expressions of the GDBG base composition functions.

This paper uses average mean error as the performance evaluation metric, calculated according to equation (6):

$$E_{\text{mean}} = \frac{1}{\text{runs} \times \text{num\_change}} \sum_{i=1}^{\text{runs}} \sum_{j=1}^{\text{num\_change}} E_{\text{last}}^{ij}$$

where runs represents the number of independent algorithm runs, num_change represents the number of changes per run, and E_{last}^{ij} represents the absolute fitness error at the j-th change during the i-th independent run, calculated as in equation (7):

$$E_{\text{last}}^{ij} = |f(x_{\text{best}}^{ij}) - f^*(x^{ij})|$$

where $f(x_{\text{best}}^{ij})$ and $f^*(x^{ij})$ denote the best solution found by the algorithm and the theoretical optimum value, respectively.

Experimental parameters include: dimension $d = 10$, change frequency $f_{\text{red}} = 10000$, change period period = 12, number of changes num_change = 60, population size $N = 10$, clone size $C = 3$, suppression threshold $\zeta = 5$, and mutation step size $\eta = 1$.

3.2 Results and Analysis

To evaluate AINMA's performance, we conducted comparative experiments on the GDBG dynamic function test suite using basic dopt-aiNet, dynamic particle swarm optimization (CPSO), orthogonal particle swarm optimization with variable relocation strategy (OLPSO-VRS), AINMA, and AINMA without tabu search. Each test function was independently run five times under identical conditions, with averaged results shown in Table 2 . Data are presented in scientific notation, with minimum error means for each scenario highlighted in bold.

Analysis of Table 2 reveals that the proposed AINMA algorithm achieves smaller error means than traditional dopt-aiNet across all 49 test scenarios, demonstrating stronger optimization capability. Compared with AINMA without tabu search, AINMA produces smaller errors in most cases, confirming that tabu search integration contributes significantly to precision improvement.

From the test function perspective, since F3's base composition function (Rastrigin) is a typical nonlinear multimodal function with large fluctuations between peaks and numerous local optima, algorithms exhibit relatively larger error means on F3 compared to other functions. Among the six test function classes, AINMA shows the most significant precision improvements on F3, F4, and F6, indicating substantially enhanced capability for solving complex problems.

From the change type perspective, the algorithm achieves relatively smaller error means under change types T1 (small changes), T4 (chaotic changes), and T6 (periodic changes with noise), demonstrating good optimization capability when problems undergo minor, chaotic, or periodic noisy changes. Overall, AINMA achieves the minimum error mean in 37 out of 49 test cases compared to dopt-aiNet, CPSO, and OLPSO-VRS. While CPSO occasionally outperforms AINMA on specific problems, and OLPSO-VRS marginally surpasses AINMA only on F3 (with negligible difference), AINMA shows substantial improvements in both convergence precision and optimization capability, exhibiting broad adaptability to various dynamic optimization problems.

Based on experimental results, convergence curves for each test function under various change types can be derived. Figures 2 [Figure 2: see original paper] through 8 [Figure 8: see original paper] illustrate convergence curves for F1-F6 under change type T1. The vertical axes represent fitness values (F1 is maximization; F2-F6 are minimization), while horizontal axes indicate environmental change counts (60 changes total, with 10 consecutive changes randomly selected for analysis).

In dynamic optimization, convergence curves typically reflect algorithmic convergence speed, response to environmental changes, and potential premature stagnation. Analyzing these curves reveals both the algorithm's adaptability to dynamic problems (whether it can promptly adjust search direction when changes occur) and its intra-change convergence speed (faster convergence enables better adaptation to rapidly changing environments).

Figure 2 shows that for F1 with 10 peaks under T1, the algorithm converges rapidly with minimal stagnation, responding promptly to environmental changes and quickly locating new global optima. Figure 3 demonstrates that increased peak count (50 peaks) does not affect convergence speed, maintaining rapid convergence despite heightened environmental complexity. Figure 4 (F2 under T1) reveals that after initial convergence from a poor starting point to the global optimum, the algorithm's fitness value does not rise significantly when environmental changes occur, indicating consistently good diversity main-

tenance. Figure 5 (F3) shows that due to the function's inherent complexity, the algorithm easily falls into local optima and adapts slowly to environmental changes initially, but optimization precision continuously improves as the algorithm gradually adapts. Figures 6-8 (F4-F6) exhibit characteristics similar to Figure 4.

In summary, AINMA inherits the diversity advantages of traditional artificial immune network algorithms while enhancing convergence speed and search precision. The algorithm responds quickly to environmental changes and initiates new searches promptly. The analysis demonstrates that AINMA exhibits strong performance for solving continuous dynamic optimization problems.

---

4 Conclusion

This paper proposes an Artificial Immune Network-based Memetic Algorithm (AINMA) for dynamic optimization, addressing the low search precision and premature convergence defects of traditional dopt-aiNet. AINMA preserves dopt-aiNet's diversity advantages while introducing a Cauchy mutation operator that alternates with traditional Gaussian mutation. Using different mutation strategies at different optimization stages leverages their respective strengths, preventing individuals from falling into local optima and enhancing global search capability. Additionally, integrating tabu search's local optimization ability into dopt-aiNet effectively improves search precision. Simulation experiments on the GDBG dynamic function test suite demonstrate that AINMA achieves substantial improvements in both search precision and convergence speed compared to its predecessor and shows clear advantages over other similar dynamic optimization algorithms.

However, the proposed AINMA algorithm exhibits limited adaptability for complex dynamic problems with numerous local optima. Future work will conduct in-depth analysis to propose improvements.

References

[1] Duan Haibin, Zhang Xiangyin, Xu Chunfang. Bio-inspired Computing [M]. Beijing: Science Press, 2011.

[2] Chen Li, Ding Lixin. Survey on dynamic optimization algorithms [J]. Journal of Wuhan University: Natural Science Edition, 2011, 57(3): 255-264.

[3] Fu Haobo, Lewis P R, Sendhoff B, et al. What are dynamic optimization problems? [C]// Proc of IEEE Congress on Evolutionary Computation. Piscataway, NJ: IEEE Press, 2014: 1550-1557.

[4] Gao Shangce, Wang Yirui, Cheng Jiujun, et al. Ant colony optimization with clustering for solving the dynamic location routing problem [J]. Applied Mathematics and Computation, 2016, 285(C): 149-173.

[5] Luo Wenjian, Yang Bin, Bu Chenyang, et al. A hybrid particle swarm optimization for high-dimensional dynamic optimization [C]// Proc of Asia-Pacific Conference on Simulated Evolution and Learning. Berlin: Springer, 2017: 981-993.

[6] Yang Xinshe, He Xingshi. Firefly algorithm: recent advances and applications [J]. International Journal of Swarm Intelligence, 2013, 1(1): 1-14.

[7] Yazdani D, Sepasmoghaddam A, Dehban A, et al. A novel approach for optimization in dynamic environments based on modified artificial fish swarm algorithm [J]. International Journal of Computational Intelligence and Applications, 2016, 15(2): 1650010.

[8] Tan Ying, Zhu Yuanchun. Fireworks Algorithm for Optimization [C]// Proc of the 1st International Conference on Advances in Swarm Intelligence. Berlin: Springer, 2010: 355-364.

[9] Liu Li, Xu Wenbo. A cooperative artificial immune network with particle swarm behavior for multimodal function optimization [C]// Proc of IEEE Congress on Evolutionary Computation. Piscataway, NJ: IEEE Press, 2008: 699-704.

[10] Zhao Hui, Li Mudong, Weng Xingwei. Distributed artificial bee colony immune algorithm for the problems of function optimization [J]. Control and Decision, 2015, 30(7): 1181-1188.

[11] De Castro L N, Timmis J. An artificial immune network for multimodal function optimization [C]// Proc of IEEE Congress on Evolutionary Computation. Piscataway, NJ: IEEE Press, 2002: 699-704.

[12] Castro L N D, Timmis J. An artificial immune network for multimodal function optimization [C]// Proc of Conference on Genetic and Evolutionary Computation. New York: ACM Press, 2005: 289-296.

[13] Zuben F J V. A dynamic artificial immune algorithm applied to challenging benchmarking problems [C]// Proc of IEEE Congress on Evolutionary Computation. Piscataway, NJ: IEEE Press, 2009: 423-430.

[14] Liu Guangyuan. Tabu search algorithm and application [M]. Beijing: Science Press, 2014.

[15] Li Zhiyong, Li Lingling, Wang Xiang, et al. Chaos artificial bee colony based on memetic framework [J]. Application Research of Computers, 2012, 29(11): 4045-4049.

[16] Zhang Xiaofei, Zhang Huoming. Improved tabu search algorithm for continuous problems [J]. Journal of China University of Metrology, 2010, 21(3): 69-74.

[17] Li Changhe, Yang Shengxiang, Nguyen T T, et al. Benchmark generator for CEC'2009 competition on dynamic optimization [J]. Sensor Letters, 2008, 11(5): 900-909.