**Journal of Computing and Electronic Information Management**
ISSN: 2413-1660 | Vol. 20, No. 1, 2026

# The Fire Seed Adaptive Diffusion and Retention Algorithm for Continuous Distributed Constraint Optimization Problems

**Meifeng Shi, Tong Fu*, †, Fuxing Ren†**

Department of Computer Science and Engineering, Chongqing University of Technology, Chongqing 400054, China

* **Corresponding author**

† **These authors contributed equally to this work**

**Abstract:** To address the challenges of proneness to local optima and population diversity decay in solving Continuous Distributed Constraint Optimization Problems (C-DCOPs), this paper proposes a novel meta-heuristic algorithm inspired by the phenomenon of spark propagation and persistence—the Fire Seed Adaptive Diffusion and Retention Algorithm (FSAD). Within this framework, a value of an individual in the population is regarded as a fire seed. By simulating the propagation, competition, and persistence processes of fire seeds, a four-stage coordinated distributed optimization framework is constructed. First, in the initialization phase, initial fire seeds are generated via controlled ran- dom sampling, and temperature parameters are configured based on problem-specific characteristics. Second, a multi-mode fire seed diffusion strategy is designed, integrating the targeted exploitation of elite fire seeds, the extensive exploration of random fire seeds, and the escape capability of reverse fire seeds. The core innovation lies in the introduction of a fire seed retention strategy, which adopts a simulated annealing-based probabilistic acceptance mechanism to ensure the persistence of superior fire seeds while dynamically maintaining population diversity. Additionally, the algorithm periodically implements a forced renewal mechanism for the worst-performing fire seeds, regularly eliminating low-vitality solutions to enhance global escape capability. Experimental results demonstrate that the FSAD algorithm outperforms traditional methods in terms of solution accuracy, convergence speed, and robustness, providing an efficient and reliable solution for C-DCOPs.

**Keywords:** C-DCOP; Adaptive diffusion; Fire seed retention; Simulated annealing; Restart mechanism.

## 1. Introduction

Distributed Constraint Optimization Problems (DCOPs) [1] serve as an effective framework for modeling distributed multi-agent systems and collaborative multi-objective optimization, holding significant research importance and practical value. In the DCOP model, a group of agents assigns values to variables under their respective control through local communication and cooperation, aiming to optimize a global objective function defined by the sum of all constraint costs [2]. Compared to centralized approaches, the distributed nature of DCOPs offers advantages in fault tolerance, privacy, and scalability, and has been successfully applied in various domains, including resource allocation [3], missile route planning [4][5], meeting scheduling [6], microgrid control [7], and task scheduling[8].

Depending on the solving objective, DCOP algorithms can be categorized into two main classes: complete algorithms and incomplete algorithms. Complete algorithms, such as SyncBB[9], ADOPT[10], BnB-ADOPT[11], and DPOP[12], aim to find the globally optimal solution but often face challenges with exponential communication or computational overhead. Subsequent research by Petcu et al. [13] proposed the MB-DPOP algorithm to reduce memory consumption, Chen et al. [14] further proposed the RMB-DPOP algorithm, and Rashik et al.[15] utilized arc-consistency techniques to shorten DPOPs runtime. Since DCOP is an NP -Hard [16] problem, incomplete algorithms, while unable to guarantee global optimality, offer superior performance in terms of communication and computational costs. Current research focuses on incomplete algorithms based on local search, such as DSA[17], MS[18],MGM[19], ACO-DCOP[20], and GDBA[21]. Frameworks like ALS[22] and PDS[23], as well as LSGA[24], have also been used to enhance the solution quality of local search-based algorithms.

However, traditional DCOPs require variables to take values from discrete domains, which limits their modeling capability for scenarios with continuous parameters, such as robotic control and continuous resource scheduling. To address this, the Continuous Distributed Constraint Optimization Problem (C-DCOP) was introduced by Stranders et al.[25], where both variables and their domains are continuous, and constraints are defined by general functions. Various algorithms have been proposed for solving C-DCOP. Methods based on the max-sum idea, like CMS and its extension HCMS[26], are limited by function representation forms or differentiability requirements. Stranders and Hoang et al. [27] proposed algorithms such as exact continuous EC-DPOP, approximate continuous AC-DPOP, CAC-DPOP, and C-DSA[28]. While these can yield high-quality solutions, they incur significant overhead. C-DSA has simple logic and low computational cost but suffers from poor solution quality. PFD[29] introduced the concept of population-based optimization to find approximate solutions. However, its effectiveness is hindered by limited search capability and a tendency to fall into local optima. Building on this, Shi et al. [30] proposed the PFD-LD algorithm, which guides individual updates using local best individuals. While this accelerates convergence, it increases message-passing overhead and compromises agent privacy due to the introduction of local decision-making. The Continuous Cooperative Constraint Approximation (C-

CoCoA) algorithm[31] can solve C-DCOP with very fast computation speed and minimal overhead, but due to its semi-greedy local search strategy, its solution quality in complex, large-scale problems is inadequate. Furthermore, the AMCGA algorithm proposed by Liao et al. [32] improves solution quality by introducing an adaptive multi-point crossover operator into a genetic algorithm, yet its convergence speed remains unsatisfactory.

Based on the aforementioned research status and challenge analysis, and to address the prevalent issues in existing C-DCOP algorithms—such as slow convergence, susceptibility to local optima, and difficulty in maintaining population diversity—this paper proposes a novel Fire Seed Adaptive Diffusion and Retention Algorithm (FSAD). The core contributions of this study are as follows:

1). A multi-mode fire seed diffusion strategy that integrates elite guidance, random exploration, and opposition-based learning. This effectively balances local exploitation and global exploration, enhancing search efficiency in continuous spaces and the ability to escape local optima.

2). A simulated annealing-based fire seed retention strategy that dynamically assesses new solutions and accepts inferior ones with a decaying probability. This preserves high-quality solutions while maintaining population diversity, mitigating premature convergence.

3). A periodic forced renewal mechanism for the worst-performing fire seeds. It periodically executes global restarts to eliminate low-quality solutions, ensuring sustained exploratory capability in later search stages.

4). A complete and reproducible C-DCOP solving framework that systematically integrates the above strategies to address the core challenges of local optima entrapment and diversity loss, offering a novel and adaptable algorithmic approach for C-DCOPs.

## 2. Continuous Distributed Constraint Optimization Problem

A C-DCOP can be defined as a tuple < A, X, D, F, α > where

$A = \{a_1, \ldots, a_n\}$ : Represents a collection of intelligent agents, where each agent may control one or more variables;

$X = \{x_1, \ldots, x_m\}$: Denotes the set of continuous variables;

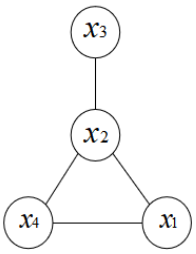
$D = \{d_1, \ldots, d_m\}$ : For a continuous range set, the variable$x_i$ takes values from the corresponding range $d_i$;

$F = \{f_1, \ldots, f_q\}$: The set of constraint cost functions, where the constraint function $f_i \in F$ satisfies$f_i : D_{i_1} \times \ldots \times D_{i_k} \to R$.

$\alpha$: A mapping function establishes the association between a continuous variable and an agent.

A feasible solution to an C-DCOP refers to the set of variable assignments $X^*$ that minimizes the sum of global constraint cost functions, and its objective function is defined as Equation (1):

$$X^* = \underset{d_i \in D_i, d_j \in D_j}{\operatorname{argmin}} \sum_{f_{ij}} f_{ij}(x_i = d_i, x_j = d_j) \quad (1)$$




(a) Constraint Graph

$$\forall xi \in X : D\, i = [-20,20]$$
$$f(x_1, x_2) = 3x_1^2 + 4x_1x_2 - x_2^2$$
$$f(x_1, x_4) = cosx_1 - x_1x_4 + x_4^2$$
$$f(x_2, x_3) = -2x_2x_3 + 5x_3^2$$
$$f(x_2, x_4) = x_2x_4 + 3x_4^2$$

(b)Cost Functions

**Fig. 1** Example of a C-DCOP

Fig.1 presents an instance model of a Continuous Distributed Constraint Optimization Problem (C-DCOP). Specifically, Fig. 1(a) shows a 4-variable constraint graph, where each edge corresponds to a constraint cost function, and the specific forms of these functions are illustrated in Fig. 1(b). All variables in this instance have a value range of [-20, 20]. The objective is to minimize the sum of all constraint cost functions, thus requiring each variable to take an appropriate value to achieve the minimal global cost.

## 3. The Fire Seed Adaptive Diffusion and Retention Algorithm

### 3.1. Overview of the Algorithm Flow

To address the challenge of balancing exploration and exploitation in C-DCOP solving, this paper proposes the Fire Seed Adaptive Diffusion and Retention Algorithm (FSAD). Inspired by the natural phenomena of spark propagation and persistence, FSAD establishes an iterative optimization framework that integrates multi-mode diffusion, probabilistic retention, and periodic restarting—designed to synergistically boost global search capability and local refinement efficiency. The algorithm follows an iterative paradigm of "Initialization – Diffusion – Retention – Update," whose overall flow is illustrated in Fig.2.

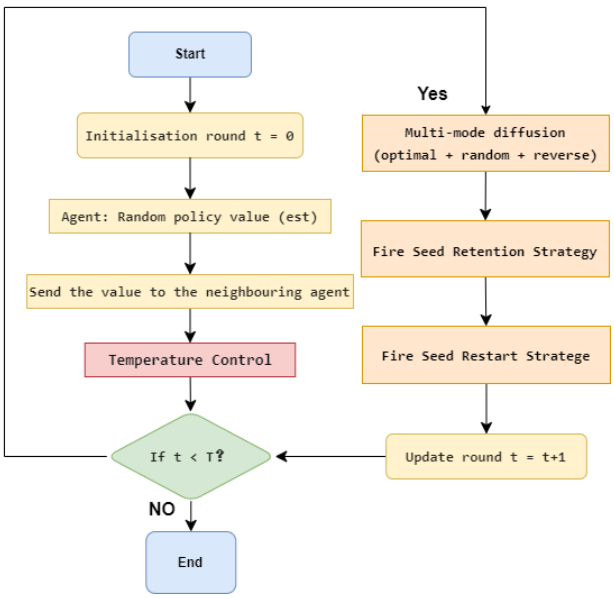



**Fig.2** Algorithm Flowchart

## 3.2. Key Steps of the Algorithm

| Algorithm: FSAD |
| --- |
| Input: Population size $P$, maximum iterations$t\,max$ , initial temperature $T0$<br>Output: Optimal fire seed $Xbest$ |
| 1.Initialization<br>2.Initialize population $\{Xi\}_{i=1}^{P}$ randomly<br>3.Evaluate fitness, sort, set $Xbest \leftarrow argmin\, f\,(X)$<br>4.$t \leftarrow 0, p \leftarrow 0, T \leftarrow T0$<br>5.Main Loop<br>6.While $t < t\,max$ :<br>7.$p \leftarrow t/t\,max$<br>8. Calculate the adaptive perturbation interval and perturbation intensity;<br>9.Dynamically adjust the current temperature T according to the iteration process;<br>11. For each ordinary fire seed$Xold$ in the population do<br>12. Record the current position and corresponding fitness of $Xold$;<br>13. Perform multi-mode diffusion operation to generate a new fire seed $Xnew$;<br>14. Calculate the fitness of $Xnew$;<br>15. Determine whether to retain $Xnew$ based on the simulated annealing criterion<br>16. (retain the optimal solution directly, accept the suboptimal solution with a certain probability);<br>17. End For<br>18. If the current iteration reaches the adaptive perturbation interval then<br>19. Reset the individual with the worst fitness in the population to a randomly generated new solution; |

P: Population size
$t\,max$ : Maximum number of iterations
$T0$: Initial temperature
$Xbest$: Optimal fire seed
$t0$: Iteration counter
$p0$: Progress parameter
X_old: Ordinary fire seed
X_new: New fire seed

### 3.2.1. Initialization

The FSAD algorithm begins by randomly generating an initial population of size P within the given solution space, where each individual constitutes a potential solution. Subsequently, all initial fire seeds are evaluated using a fitness function to compute their global fitness value, as defined in Equation (2).

$$P_i \cdot fitness = \sum_{f_i \in F} f_i(P_i, P_{n_i}) \quad (2)$$

The optimal fire seed is identified based on fitness ranking, corresponding to the best individual $Xbest$ that ranks first. Concurrently, the temperature parameter $T0$ for the simulated annealing mechanism is initialized, calculated according to the following Equation (3):

$$T0 = \frac{(d_u - d_l) \times 10}{1 + ln(1 + t0)} \quad (3)$$

Here, $t0$ represents the current iteration count and is initialized to 0. Utilizing the properties of the logarithmic function, this formulation maintains the initial temperature at a level approximately ten times the span of the solution space. This ensures a high probability of accepting suboptimal solutions during the early iterations, thereby creating favorable conditions for maintaining population diversity. Additionally, the iteration progress parameter is initialized according to Equation (4), with $p0$starting at a value of 0. This establishes a baseline for the dynamic adjustments of subsequent adaptive strategies.

$$p0 = \frac{t0}{t\,max} \quad (4)$$

Through these steps, the initialization phase constructs an initial set containing both optimal and ordinary fire seeds. It also configures temperature and progress parameters tailored to the algorithms requirements for exploration and exploitation. This establishes a stable and promising starting point for the subsequent multi-mode diffusion phase.

### 3.2.2. Random Fire Seed Diffusion

The random fire seed diffusion strategy is designed to preserve global exploratory breadth and is triggered independently with probability$pr$. For a selected ordinary fire seed, a new solution is generated randomly within the solution space according to Equation (5).

$$Xnew = dl + U(0,1) \times (du - dl) \quad (5)$$

This strategy does not rely on existing solution information. By periodically injecting entirely new solutions, it effectively prevents the algorithm from becoming trapped in local optima and significantly expands the search range throughout the iterative process.

### 3.2.3. Optimal Fire Seed Diffusion

The optimal fire seed diffusion strategy focuses on enhancing local exploitation. Its activation probability increases dynamically with iteration progress. Guided by the optimal fire seed, this strategy generates new solutions via Gaussian perturbation, as defined in Equation (6):

$$Xnew = Xbest + N(0,1) \times \sigma \quad (6)$$

Here,$N(0,1)$ follows a standard normal distribution. The perturbation intensity $\sigma$ decreases as the progress parameter $p0$ increases, scaling down from 0.5 to 0.05 times the solution space span. This achieves a smooth transition from "broad exploration around the elite" to "refined local optimization," ensuring deep exploitation of promising regions while preventing the loss of diversity caused by excessive concentration.

### 3.2.4. Reverse Fire Seed Diffusion

The reverse fire seed diffusion strategy leverages the symmetry of the solution space to expand search dimensions. A new solution is generated based on the symmetrical point of the current solution, as defined in Equation (7):

$$Xnew = (dl + du - Xold) + N\,(0,1) \times \sigma \quad (7)$$

Here, $dl + du - Xold$ is the symmetrical point of $Xold$ about the center of the solution space. The coefficient $\sigma$ is set to induce a weaker perturbation magnitude compared to the optimal fire seed diffusion strategy. This preserves the exploratory value of symmetrical points while preventing excessive deviation from the current search direction. By probing spatial regions complementary to existing solutions, this strategy effectively enhances population diversity. It is particularly capable of providing new search directions when the algorithm converges to local optima.

### 3.2.5. Fire Seed Retention Mechanism

The fire seed retention mechanism is a probabilistic acceptance strategy based on simulated annealing. It serves as a critical component of the FSAD algorithm for filtering high-quality solutions while maintaining population diversity. Its core function is to selectively retain new fire seeds generated

in the multi-mode diffusion phase using the simulated annealing acceptance criterion. This ensures the stable propagation of superior solutions while allowing potentially promising, yet suboptimal, solutions to survive.

Specifically, the mechanism involves a two-step decision logic. First, the newly generated fire seed is evaluated for fitness, and its fitness difference from the original seed $Xold$ is calculated using Equation (8):

$$\Delta f = f(Xnew) - f(Xold) \quad (8)$$

A preliminary screening is then performed based on$\Delta f$. If $\Delta f < 0$, indicating the new fire seed is superior, it is directly accepted as $Xnew$.For suboptimal solutions that fail this initial screening , the mechanism applies a probabilistic acceptance strategy from simulated annealing. The acceptance probability is calculated using Equation (9).

$$P(accept) = exp(-|\Delta f| / T0) \quad (9)$$

Here, $|\Delta f|$ represents the absolute value of the fitness difference, and $T0$ is the current temperature. The core principle of this formula is that the probability of accepting a suboptimal solution increases when its fitness loss is smaller (i.e., $|\Delta f|$ is smaller) or when the current temperature $T0$ is higher.

Coupled with the characteristic that $T0$ decreases as the iteration count $t0$ increases—maintaining a relatively high value initially when $t0$ is small and then gradually decaying through the logarithmic function—this mechanism dynamically regulates the acceptance of suboptimal solutions. In the early iterations, the higher temperature allows more suboptimal solutions to be retained, which helps maintain population diversity and encourages exploration of a broader solution space. As iterations progress and the temperature decreases, the acceptance probability contracts, permitting only suboptimal solutions with minor fitness losses to survive. This progressively guides the algorithm toward refined convergence near local optima.

#### 3.2.6. Fire Seed Restart Strategy

The dynamic update of fire seeds is a core regulatory mechanism in the FSAD algorithm, designed to maintain population vitality and prevent search stagnation. It operates by periodically resetting the worst-performing fire seed in the population, thereby eliminating persistently poor solutions and injecting new exploratory potential.

The update is triggered at dynamic intervals. The interval length increases linearly from 5 to 50 generations as the algorithm progresses from initial exploration to final refinement. This design balances competing needs: shorter intervals in early phases quickly purge inferior solutions to accelerate evolution, while longer intervals in later phases minimize disruption to convergence around promising regions.

Upon triggering, the worst fire seed is replaced by a completely new, randomly generated solution within the global search space. This targeted reset prevents resource wastage on poor solutions and introduces novel information from unexplored areas. Compared to the general random diffusion strategy, this mechanism is more precise, forcibly renewing only the proven weakest individual to enhance diversity with minimal impact on the overall population structure.

# 4. Complexity Analysis of the Algorithm

The time complexity of the FSAD algorithm is analyzed as follows. Let *K* be the population size and *T* the maximum number of iterations.

**Initialization Phase:** This involves generating the initial population (*O(K)*), evaluating all individual fitness values (*O(K)*), and sorting them to identify the optimal fire seed (*O(K log K)*). The overall complexity for this phase is therefore *O(K log K)*.

**Iteration Phase:** Each iteration consists of several core operations: updating parameters (*O(1)*), executing multi-mode diffusion for *K-1* fire seeds (*O(K)*), performing fire seed retention checks (*O(K)*), updating the worst fire seed (*O(1)*), and finally re-evaluating and sorting the population (*O(K log K)*). The dominant cost per iteration is *O(K log K)*.

Thus, for *T* iterations, the total time complexity is *O(T · K log K)*. The initialization cost is comparatively negligible, leading to a final overall complexity of **O(T · K log K)**.

# 5. Experimental Results and Analysis

## 5.1. Benchmark Problems

**Random Graphs:** In the experiments, the number of agents varied from 50 to 100 in increments of 10 to generate test networks of different scales. Edge densities of p = 0.1 and p = 0.6 were used to represent low-density and high-density problems, respectively.

**Scale-free Networks:** Network topologies with power-law characteristics were generated based on the Barabási-Albert (BA) model. The process began with a connected graph of 10 agents as the initial structure. In each subsequent step, a new agent was added and connected to the 7 existing agents with the highest degrees. The connection probability for the new agent was proportional to the degree of existing nodes, forming a scale-free network through this preferential attachment mechanism. The number of agents was similarly configured in the range of 50 to 100 with an increment of 10.

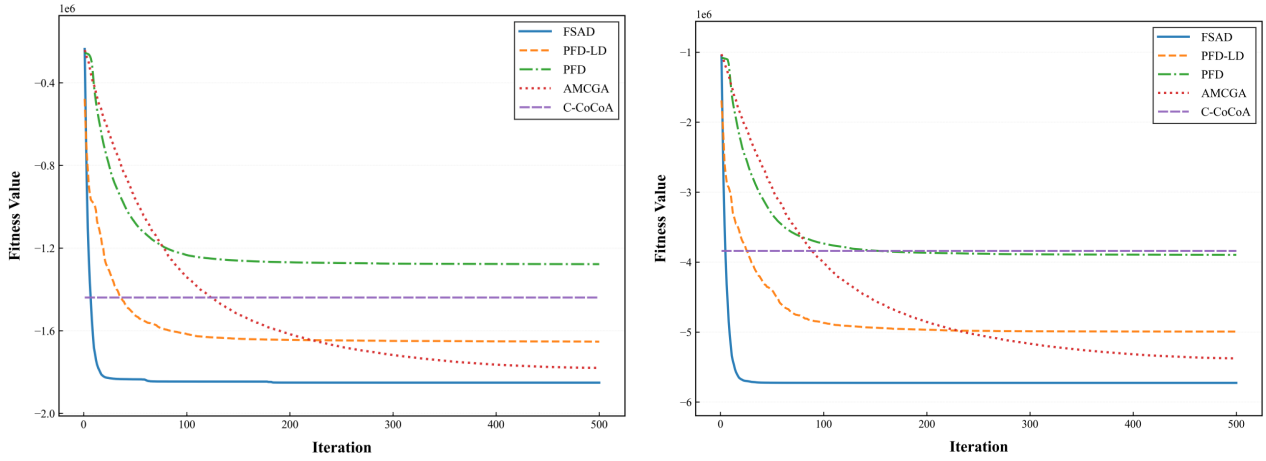
**Random Trees:** Agent counts were set from 50 to 100 in steps of 10, constructing tree-structured networks of various sizes to evaluate the algorithms performance under tree topologies.

**Small-world Networks:** The construction started with a ring network where each node was connected to its J nearest neighbors (J=6 in this experiment). Edges were then randomly rewired with a probability P=0.5, strictly avoiding duplicate edges and self-loops during the process. The number of agents ranged from 50 to 100 with an increment of 10 to investigate the algorithms effectiveness in small -world network environments.

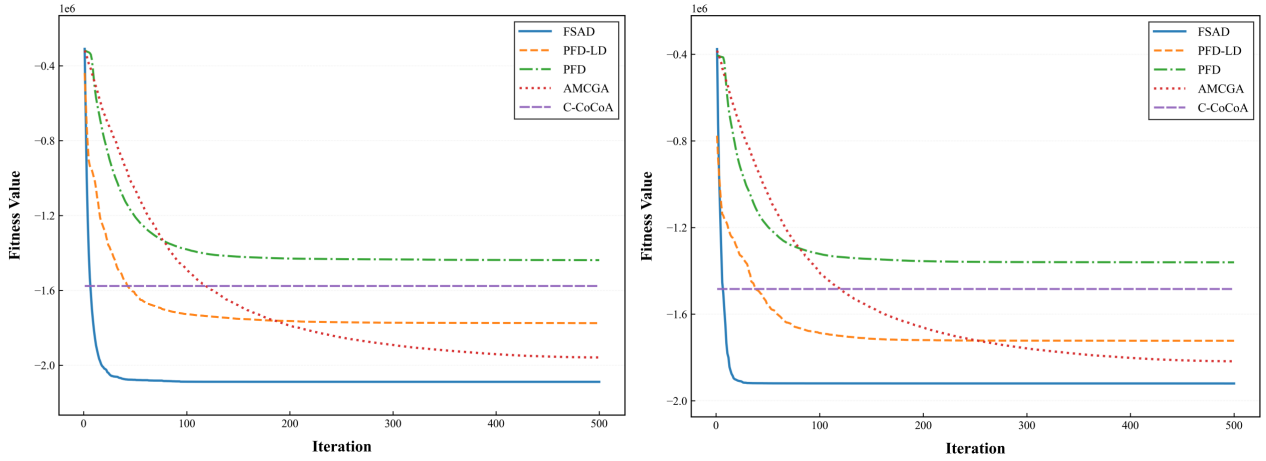
## 5.2. Comparative Experiments of Algorithms

The experiments focus on Distributed Constraint Optimization Problems (DCOPs). The proposed FSAD algorithm is evaluated against four comparative algorithms: PFD-LD, PFD, AMCGA, and C-CoCoA. Performance is assessed from three perspectives: convergence speed, final solution quality, and algorithm stability, by analyzing the evolution of fitness values over iterations.

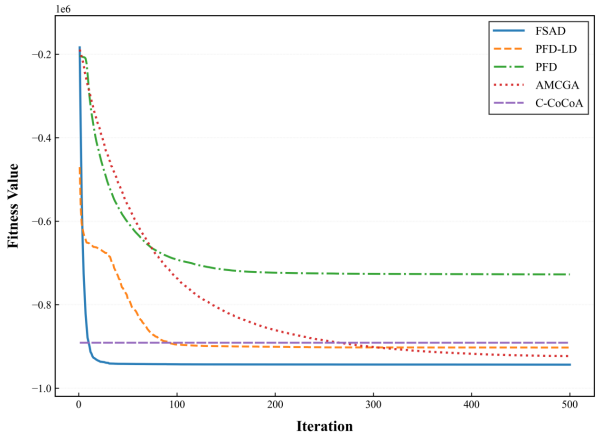
To account for the inherent randomness of incomplete search algorithms, rigorous statistical measures are adopted. For each benchmark problem class, 50 random instances are generated. Each instance is independently executed 30 times, with average results reported to ensure generalizability and robustness. The experimental outcomes are summarized as follows:




(a) Sparse Random Graph (b) Dense Random Graph




(c) Scale-Free Network (d) Small-World Network




(e) Random Tree

**Fig. 3** Convergence curves of FSAD and comparative algorithms

Fig.3 demonstrates FSAD superior convergence performance, achieving stable solutions within 50–100 iterations—significantly faster than the 100–200 iterations required by the other algorithms. It exhibits the most rapid initial fitness decrease, promptly escaping poor solution regions.In terms of final solution quality, FSAD attains the lowest fitness values across all tests, outperforming all other algorithms. PFD-LD and PFD form the second-best tier, while AMCGA and C-CoCoA yield the highest and thus poorest fitness values.Regarding stability, FSAD maintains minimal fitness fluctuation after convergence, indicating strong robustness. In contrast, the other algorithms exhibit minor but noticeable variations.

In summary, FSAD excels in convergence speed, solution quality, and stability, validating its effectiveness for solving distributed constraint optimization problems (DCOPs).

To quantitatively validate the aforementioned performance advantages of FSAD, Table 1 reports the relative improvement percentages of FSAD over the comparative algorithms across five typical benchmark problems for DCOPs.

**Table 1.** Improvements of FSAD over comparative algorithms on benchmark problems

| Benchmark Problems | PFD-LD | PFD | AMCGA | C-CoCoA |
|---|---|---|---|---|
| Sparse Random Graph | 12.01% | 44.90% | 4.01% | 28.66% |
| Dense Random Graph | 14.69% | 46.97% | 6.51% | 49.16% |
| Scale-Free Network | 17.67% | 45.19% | 6.66% | 32.50% |
| Random Tree | 4.54% | 29.72% | 2.23% | 5.90% |
| Small-World Network | 11.45% | 41.12% | 5.64% | 29.44% |

# 6. Conclusion

This study proposes the Fire Seed Adaptive Diffusion and Retention Algorithm (FSAD). The algorithm effectively coordinates global exploration and local exploitation by integrating an optimal fire seed guidance mechanism, a multi-mode adaptive diffusion strategy, a simulated annealing retention mechanism, and a periodic dynamic update mechanism. Experimental results demonstrate that FSAD consistently achieves high-quality solutions and exhibits superior convergence performance across diverse test problems. Its adaptive strategy design offers a novel approach for solving complex continuous distributed constraint optimization problems. Future work may extend the algorithms application to multi -objective optimization problems or incorporate hybrid strategies with other intelligent optimization algorithms to enhance its performance in high-dimensional, highly-constrained scenarios.

# Declarations


• Funding

This work is supported by the Youth Project of Science and Technology Research Pro- gram of Chongqing Education Commission of China (NO. KJQN202401101).


• Conflict of interest/Competing interests (check journal-specific guidelines for which heading to use)

The authors declare there are no competing interests.

• Ethics approval and consent to participate Not applicable.

• Consent for publication

Written informed consent for publication was obtained from all participants.

• Data availability

Experimental data supporting the findings of this study are available from the correspond- ing author upon reasonable request.

• Materials availability Not applicable.

• Code availability

The code supporting the conclusions of this study will be made publicly available upon publication of the final version of the article.

• Author contribution

Meifeng Shi conceived and designed the experiments, analyzed the data and created a flow diagram , authored and reviewed drafts of the article, and approved the final draft. Tong Fu performed the experiments, prepared other figures and tables, analyzed the data, authored drafts of the article, and

# References


[1] He C. Research on Search Algorithms for Solving Distributed Constraint Optimization Problems [D]. Chongqing: Chongqing University, 2016.

[2] Liao X. Algorithm Research on Continuous Distributed Constraint Optimization Problems Based on Swarm Intelligence [D]. Chongqing: Chongqing University of Technology, 2023. DOI: 10.27753/d.cnki.gcqgx.2023.000994.

[3] Enembreck F, Barthes J P A. Distributed constraint optimization with MULBS: A case study on collaborative meeting scheduling [J]. Journal of Network and Computer Applications, 2012, 35(1): 164-175.

[4] Farinelli A, Rogers A, Jennings N R. Agent-based decentralised coordination for sensor networks using the max-sum algorithm [J]. Autonomous Agents and Multi-Agent Systems, 2014, 28: 337-380.

[5] Muldoon C, O'Hare G M, O'Grady M J, et al. Distributed constraint optimisation for resource limited sensor networks [J]. Science of Computer Programming, 2013, 78(5): 583-593.

[6] Sultanik E A, Modi P J, Regli W C. On modeling multiagent task scheduling as a distributed constraint optimization problem [C]//Proceedings of the 20th International Joint Conference on Artificial Intelligence. New York: ACM, 2007: 1531-1536.

[7] Ma R, Jin Y, Liu M C. Bi-level optimal configuration of distributed wind and photovoltaic generations in active distribution network based on chance constrained programming [J]. Transactions of China Electrotechnical Society, 2016, 31(3): 145-154.

[8] Barths A, Enembreck F. Distributed constraint optimization with MULBS: a case study on collaborative meeting scheduling [J]. Journal of Network and Computer Applications, 2018, 30(6): 64-68.

[9] Hirayama K, Yokoo M. Distributed partial constraint satisfaction problem [C]//Proc of the 3rd International Conference on Principles and Practice of Constraint Programming, 1997: 222-236.

[10] Modi P J, Shen W M, Tambe M, et al. ADOPT: Asynchronous distributed constraint optimization with quality guarantees [J]. Artificial Intelligence, 2005, 161(1): 149-180.

[11] Yeoh W, Felner A, Koenig S. BnB-ADOPT: An asynchronous branch-and-bound DCOP algorithm [J]. Journal of Artificial Intelligence Research, 2010, 38: 85-133.

[12] Petcu A, Faltings B. A scalable method for multiagent constraint optimization [C]//Proceedings of the 19th International Joint Conference on Artificial Intelligence. New York: ACM, 2005: 266-271.

[13] Petcu A, Faltings B. Mb-dpop: A new memory-bounded algorithm for distributed optimization [C]//Proc of the 20th

International Joint Conference on Artificial Intelligence, 2007: 1452-1457.

[14] Chen Z, Zhang W, Deng Y, et al. RMB-DPOP: Refining MB-DPOP by Reducing Redundant Inference [C]//Proc of the 19th International Conference on Autonomous Agents and Multiagent Systems, 2020: 249-257.

[15] Rashik M, Rahman M M, Khan M M, et al. Speeding up distributed pseudo-tree optimization procedures with cross edge consistency to solve DCOPs [J]. Appl Intell, 2021, 51: 1733-1746.

[16] Fursin G, Cohen A. A practical method for quickly evaluating program optimizations [C]//Proc of the 1st International Conference on High Performance Embedded Architectures & Compilers (HiPEAC 2005), 2005: 29-46.

[17] Zhang W, Wang G, Xing Z, et al. Distributed stochastic search and distributed breakout: properties, comparison and applications to constraint optimization problems in sensor networks [J]. Artificial Intelligence, 2005, 161(1-2): 55-87.

[18] Farinelli A, Rogers A, Petcu A, et al. Decentralised coordination of low-power embedded devices using the max-sum algorithm [C]//Proceedings of the 7th International Joint Conference on Autonomous Agents and Multi-Agent Systems. Estoril: AAMAS Press, 2008: 639-646.

[19] Maheswaran R T, Pearce J P, Tambe M. A family of graphical-game-based algorithms for distributed constraint optimization problems [J]. Coordination of Large-Scale Multi Agent Systems, 2006: 127-146.

[20] Chen Z Y, Wu T F, Deng Y C, et al. An ant based algorithm to solve distributed constraint optimization problems [C]//Proceedings of the 32nd AAAI Conference on Artificial Intelligence. New Orleans: AAAI Press, 2018: 4653-4661.

[21] Okamoto S, Zivan R, Nahon A. Distributed Breakout: Beyond Satisfaction [C]//Proc of the 25th International Joint Conference on Artificial Intelligence, 2016: 447-453.

[22] Zivan R. Anytime local search for distributed constraint optimization [C]//Proc of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems, 2008: 1449-1452.

[23] Chen Z, Yu Z, He J, et al. A partial decision scheme for local search algorithms for distributed constraint optimization problems [C]//Proc of the 16th International Conference on Autonomous Agents and Multiagent Systems, 2017: 187-194.

[24] Chen Z, Liu L, He J, et al. A genetic algorithm based framework for local search algorithms for distributed constraint optimization problems [J]. Autonomous Agents and Multi-Agent Systems, 2020, 34: 1-31.

[25] Stranders R, Farinelli A, Rogers A, et al. Decentralised control of continuously valued control parameters using the max-sum algorithm [C]//Proc of the 8th International Conference on Autonomous Agents and Multiagent Systems, 2009: 601-608.

[26] Voice T, Stranders R, Rogers A, et al. A hybrid continuous max-sum algorithm for decentralised coordination [C]//Proceedings of the 19th European Conference on Artificial Intelligence. Lisbon: IOS Press, 2010: 61-66.

[27] Hoang K D, Yeoh W, Yokoo M, et al. New algorithms for continuous distributed constraint optimization problems [C]//Proc of the 19th International Conference on Autonomous Agents and Multiagent Systems, 2020: 502-510.

[28] Khoi D H, Yeoh W, Yokoo M, et al. New algorithms for continuous distributed constraint optimization problems [C]//Proceedings of the 19th International Conference on Autonomous Agents and Multi-Agent Systems. Auckland: Springer Press, 2020: 502-510.

[29] Choudhury M, Mahmud S, Khan M M. A particle swarm based algorithm for functional distributed constraint optimization problems [C]//Proceedings of the 34th AAAI Conference on Artificial Intelligence. New York: AAAI Press, 2020: 7111-7118.

[30] Shi M, Liao X, Chen Y. A particle swarm with local decision algorithm for functional distributed constraint optimization problems [J]. International Journal of Pattern Recognition and Artificial Intelligence, 2022, 36(12): 2259025.

[31] Amit S, Moumita C, Md M K. A local search based approach to solve continuous DCOPs [C]//Proceedings of the 20th International Conference on Autonomous Agents and Multi-Agent Systems. Richland: Springer Press, 2021: 1127-1135.

[32] Liao X, Shi M F, Chen Y. Adaptive multi-point crossover genetic algorithm for solving continuous distributed constraint optimization problems [J]. CAAI Transactions on Intelligent Systems, 2023, 18(4): 793-802.