Advances in Intelligent Systems and Computing 834

Jeng-Shyang Pan
Jerry Chun-Wei Lin
Bixia Sui
Shih-Pang Tseng *Editors*

Genetic and Evolutionary Computing

Proceedings of the Twelfth International Conference on Genetic and Evolutionary Computing, December 14–17, Changzhou, Jiangsu, China

Springer

Advances in Intelligent Systems and Computing

Volume 834

The series "Advances in Intelligent Systems and Computing" contains publications on theory, applications, and design methods of Intelligent Systems and Intelligent Computing. Virtually all disciplines such as engineering, natural sciences, computer and information science, ICT, economics, business, e-commerce, environment, healthcare, life science are covered. The list of topics spans all the areas of modern intelligent systems and computing such as: computational intelligence, soft computing including neural networks, fuzzy systems, evolutionary computing and the fusion of these paradigms, social intelligence, ambient intelligence, computational neuroscience, artificial life, virtual worlds and society, cognitive science and systems, Perception and Vision, DNA and immune based systems, self-organizing and adaptive systems, e-Learning and teaching, human-centered and human-centric computing, recommender systems, intelligent control, robotics and mechatronics including human-machine teaming, knowledge-based paradigms, learning paradigms, machine ethics, intelligent data analysis, knowledge management, intelligent agents, intelligent decision making and support, intelligent network security, trust management, interactive entertainment, Web intelligence and multimedia.

The publications within "Advances in Intelligent Systems and Computing" are primarily proceedings of important conferences, symposia and congresses. They cover significant recent developments in the field, both of a foundational and applicable character. An important characteristic feature of the series is the short publication time and world-wide distribution. This permits a rapid and broad dissemination of research results.

**** Indexing: The books of this series are submitted to ISI Proceedings, EI-Compendex, DBLP, SCOPUS, Google Scholar and Springerlink ****

More information about this series at http://www.springer.com/series/11156

Jeng-Shyang Pan · Jerry Chun-Wei Lin ·
Bixia Sui · Shih-Pang Tseng
Editors

Genetic and Evolutionary Computing

Proceedings of the Twelfth International Conference on Genetic and Evolutionary Computing, December 14–17, Changzhou, Jiangsu, China

Springer

Editors
Jeng-Shyang Pan
Fujian Provincial Key Laboratory of Big Data Mining and Applications
Fujian University of Technology
Fuzhou, Fujian, China

Bixia Sui
Changzhou College of Information Technology
Changzhou, Jiangsu, China

Jerry Chun-Wei Lin
Department of Computing, Mathematics, and Physics
Western Norway University of Applied Sciences
Bergen, Norway

Shih-Pang Tseng
College of Smart Living and Management
Tajen University
Pingtung, Taiwan

ISSN 2194-5357 ISSN 2194-5365 (electronic)
Advances in Intelligent Systems and Computing
ISBN 978-981-13-5840-1 ISBN 978-981-13-5841-8 (eBook)
https://doi.org/10.1007/978-981-13-5841-8

Library of Congress Control Number: 2019930370

This Springer imprint is published by the registered company Springer Nature Singapore Pte Ltd.
The registered company address is: 152 Beach Road, #21-01/04 Gateway East, Singapore 189721, Singapore

Conference Organization

Honorary Chairs

Yong Zhou, Changzhou College of Information Technology, China
Jeng-Shyang Pan, Fujian University of Technology, China
Jhing-Fa Wang, Tajen University, Taiwan
Xin Tong, Fujian University of Technology, China

Advisory Committee Chairs

Tzung-Pei Hong, National University of Kaohsiung, Taiwan
Bixia Sui, Changzhou College of Information Technology, China

General Chairs

Jerry Chun-Wei Lin, Western Norway University of Applied Sciences, Norway
Pyke Tin, University of Computer Studies, Myanmar
Vaclav Snasel, VSB-Technical University of Ostrava, Czech Republic

Program Chairs

I-Hsin Ting, National University of Kaohsiung, Taiwan
Lyuchao Liao, Fujian University of Technology, China
Philippe Fournier-Viger, Harbin Institute of Technology (Shenzhen), China

Shih-Pang Tseng, Tajen University, Taiwan
Shu-Chuan Chu, Flinders University, Australia

Local Organization Chair

Jianhui Zhao, Changzhou College of Information Technology, China

Electronic Media Chairs

Shi-Jian Liu, Fujian University of Technology, China
Thi-Thi Zin, University of Miyazaki, Japan

Publication Chair

Chien-Ming Chen, Harbin Institute of Technology (Shenzhen), China

Finance Chair

Jui-Fang Chang, National Kaohsiung University of Sciences and Technology, Taiwan

Program Committee Members

An-chao Tsai, Tajen University, Taiwan
Ashish Sureka, Ashoka University, India
Bay Vo, Ho Chi Minh City University of Technology, Vietnam
Bo Zhang, Changzhou Vocational Institute of Mechatronic Technology, China
Chao Huang, Southeast University, China
Chien-Ming Chen, Harbin Institute of Technology (Shenzhen), China
Chuan Zhu, Changzhou College of Information Technology, China
Chun-Hao Chen, Tamkang University, Taiwan
Gang Sun, Nanjing College of Information Technology, China
Hong Zhu, Changzhou Vocational Institute of Textile and Garment, China
Hui Wang, Nanchang Institute of Technology, China
Hsun-Hui Huang, Tajen University, Taiwan

Hua Lou, Changzhou College of Information Technology, China
I-Hsin Ting, National University of Kaohsiung, Taiwan
Jean Lai, Hong Kong Baptist University, Hong Kong
Jerry Chun-Wei Lin, Western Norway University of Applied Sciences, Norway
Ji Zhang, University of Southern Queensland, Australia
Jimmy Ming-Tai Wu, Shandong University of Science and Technology, China
Kawuu W. Lin, National Kaohsiung University of Sciences and Technology, Taiwan
Lan Cao, Changzhou College of Information Technology, China
Ming-Jin Li, Huaian Vocational College of Information Technology, China
Ming-Wei Liu, Tianjin Bohai Vocational Technical College, China
Pei-Wei Tsai, Swinburne University of Technology, Australia
Qian Yin, Changzhou College of Information Technology, China
Shouyun Chu, Changzhou Vocational Institute of Light Industry, China
Thi-Thi Zin, University of Miyazaki, Japan
Tsu-Yang Wu, Fujian University of Technology, China
Tzung-Pei Hong, National University of Kaohsiung, Taiwan
Wei-Min Chen, Yili Vocational and Technical College, China
Wen-Quan Zeng, Guangdong University of Science and Technology, China
Wensheng Gan, Harbin Institute of Technology (Shenzhen), China
Xiang-Yang Liu, Nanjing University, China
Xiao-Hua Zou, Changzhou College of Information Technology, China
Xiao-Jun Fan, Shanghai University, China
Xiao-Jun Wu, Jiangnan University, China
Xiao-Yan Tang, Changzhou College of Information Technology, China
Xin-Chang Zhang, Nanjing University of Information Science and Technology, China
Xin-Hua Li, Shandong Polytechnic, China
Yi-Chun Lin, Tajen University, Taiwan
Yong Zhou, Changzhou Vocational Institute of Engineering, China
Yong-Hua Xie, Nanjing Institute of Industry Technology, China
Yuh-Chung Lin, Tajen University, Taiwan
Yun-Fei Wu, Changzhou College of Information Technology, China
Yu-Rong Ren, Changzhou University, China
Zai-Wu Gong, Nanjing University of Information Science and Technology, China
Ze-Zhong Xu, Changzhou Institute of Technology, China
Zhong-Ping Qian, Changzhou College of Information Technology, China

Preface

This volume composes the Proceedings of the Twelfth International Conference on Genetic and Evolutionary Computing (ICGEC 2018), which was hosted by Changzhou College of Information Technology and was held in Changzhou, Jiangsu, China on December 14–17, 2018. ICGEC 2018 was technically cosponsored by Changzhou College of Information Technology (China), Fujian Provincial Key Lab of Big Data Mining and Applications (Fujian University of Technology, China), National Demonstration Center for Experimental Electronic Information and Electrical Technology Education (Fujian University of Technology, China), Tajen University (Taiwan), National University of Kaohsiung (Taiwan), Shandong University of Science and Technology (China), Western Norway University of Applied Sciences (Norway), and Springer. It aimed to bring together researchers, engineers, and policymakers to discuss the related techniques, to exchange research ideas, and to make friends. Eighty-one excellent papers were accepted for the final proceeding. Six plenary talks were kindly offered by: Prof. James, Jhing-Fa Wang (President of Tajen University, Taiwan, IEEE Fellow), Prof. Zhigeng Pan (Hangzhou Normal University, China), Prof. Xiudong Peng (Chengzhou College of Information Technology, China), Prof. Jiuyong Li (University of South Australia, Australia), Prof. Philippe Fournier-Viger (Harbin Institute of Technology (Shenzhen), China), and Prof. Peter Peng (University of Calgary, Canada). We would like to thank the authors for their tremendous contributions. We would also like to express our sincere appreciation to the reviewers, Program Committee members, and the Local Committee members for making this conference successful.

Fuzhou, China — Jeng-Shyang Pan
Bergen, Norway — Jerry Chun-Wei Lin
Changzhou, China — Bixia Sui
Pingtung, Taiwan — Shih-Pang Tseng
December 2018

Contents

Part I
Nature Inspired Constrained Optimization

A New Advantage Sharing Inspired Particle Swarm Optimization Algorithm

Lingping Kong and Václav Snášel

Abstract Particle swarm optimization algorithm is a widely used computational method for optimizing a problem. This algorithm has been applied to many applications due to its easy implementation and few particles required. However, there is a big problem with the PSO algorithm, all the virtual particles converged to a point which may or may not be the optimum. In the paper, we propose an improved version of PSO by introducing the idea of advantage sharing and pre-learning walk mode. The advantage sharing means that the good particles share their advantage attributes to the evolving ones. The pre-learning walk mode notices one particle if it should continue to move or not which uses the feedback of the last movement. Two more algorithms are simulated as the comparison methods to test Benchmark function. The experimental results show that our proposed scheme can converge to a better optimum than the comparison algorithms.

Keywords Particle swarm optimization · Advantage sharing · Benchmark function

1 Introduction

A collective behavior of natural or artificial is the concept of Swarm intelligence (SI) [1]. The agents follow very simple rules, and although there is no centralized control structure guiding how members should behave, communications between such agents converge to the global behavior. Swarm Intelligence-based techniques can be used in a number of applications. Swarm Intelligence-based algorithms include Cat Swarm Optimization [2], Genetic algorithm (GA), Artificial Bee Colony Optimization [3],

L. Kong · V. Snášel (✉)
Faculty of Electrical Engineering and Computer Science, VSB-Technical University of Ostrava, Ostrava, Czech Republic
e-mail: vaclav.snasel@vsb.cz

L. Kong
e-mail: konglingping2007@163.com

J.-S. Pan et al. (eds.), *Genetic and Evolutionary Computing*,
Advances in Intelligent Systems and Computing 834,
https://doi.org/10.1007/978-981-13-5841-8_1

Particle swarm optimization (PSO) [4], Shuffle frog leaping algorithm (SFLA) [5], and so on.

PSO is originally proposed by Kennedy and Eberhart for simulating social behavior, then this algorithm was simplified and was observed to be performing optimization. A number of candidate particles represent problem solutions are simulated and iteratively improved toward the best. The most common type of implementation defines the updating mode with two equations, velocity updating, and position updating. There are many variants of PSO [6], some researchers seek ways to simplify the algorithm, some versions improve the updating mode by adding extra features to it [7], some applications combine PSO with other algorithms attributes to solve problems [8].

Ahmad Nickabadi is one of the authors who studied the parameter decision of updating equation. The author analyzed and discussed three main groups of choosing inertia weight value, constant, time-varying, and adaptive inertia weights. Except for that, the author proposed a new approach with adaptive inertia weight parameter, the populations situation can be reflected based on the feedback parameter which uses the success rate of the particles [9]. Moradi [10] combines the genetic algorithm and particle swarm optimization for location decision use on a distribution system. The author uses the combination strategy to minimize network power losses and improve the voltage stability, and a good performance is demonstrated by carrying out this strategy on 33 and 69 bus systems. A hybrid algorithm used a fuzzy adaptive particle swarm optimization (FAPSO) and Nelder–Mead simplex search is proposed by Taher Niknam [11]. This hybrid algorithm completes the local search through the Nelder–Mead process and accomplishes the global search by FAPSO. The author verified the hybrid algorithm by testing it on two typical systems consisting of 13 and 40 thermal units. Cheng [12] proposed a social learning particle swarm optimization (SLPSO). This algorithm adopts a dimension-dependent parameter to relief the parameter settings and it also uses demonstrators in the current swarm to guide the particle evolving, which unlike the classical PSO variants. The experimental results show the algorithm performs well on low-dimensional and high-dimensional problems.

The rest of paper is organized as follows: Sect. 2 introduces the disadvantage of PSO algorithm and presents the idea of advantage sharing, then proposes the AS-PSO algorithm. Section 3 gives the results of comparison experiments tested on Benchmark function. Section 4 concludes the paper.

2 Advantage Sharing Particle Swarm Optimization

The particle swarm optimization (PSO) is a computational method and also a population-based stochastic algorithm. PSO optimizes a problem by updating candidate solutions iteratively, which also means that there are no selection and random walk during the running process. The candidate members are active based on its local best-known location over the individuals position and velocity, and their movements are influenced by the best one known location in searching space. Even PSO can search very large spaces of candidate solutions, but it does not guarantee an

optimum solution. In other words, the PSO algorithm may converge to a local optimum. So this paper proposes an improved version of PSO with the idea of advantage sharing and pre-learning walk, called AS-PSO.

In the advantage sharing, one individual is guided toward several the best-known positions instead of one best position, the common properties of those better positions could be the characteristics of the optimum location in a big probability, and advantage sharing decreases the occurrence of trapping into a local optimum. Before the population updating, each particle will be set up a movement flag, which labels the moving direction of its last step, and before each particles move, there is also a maximum steps threshold holds the particles moving times. The movement flag works as the pioneer wizard, it tells the particles whether the last movement gets better or not. If one particle moves in a direction and gets improvement, then it might get more improvement after moving along this direction. AS-PSO algorithm consists of a set of virtual particle population. During the searching process, every particle will evolve based on the basic PSO operation. The different points are in three parts, first before an operation, each particle will be given a movement flag, if the previous movement is good, then continue to move until it reaches the biggest moving threshold. Second, if the first step searching does not improve its current status, then goes to the advantage sharing operation. Third, adding a small proportional random walk operation. The process of AS-PSO in detail is introduced as follows:

2.1 Steps of the AS-PSO

Step 0. Setting-up phase. Define a problem space with D dimensions, the virtual population P with N particles. The evolving iteration times is t. define the final solution variable as g_{best}, and $threshold_1$ is the maximum step number that one individual can move, $threshold_2$ is the probability of random walk for a particle. Set $MC = 0$ (Moving Count, MC).

Step 1. Initialize population $P = \{p_1, p_2, \ldots, p_N\}$, and each particle has its random position and velocity within the search space. For particle j, its position can be labeled as $X_j = \{x_1, x_2, \ldots, x_D\}$, the velocity is $V_j = \{v_1, v_2, \ldots, v_D\}$. Other than that, each particle has p_h value, it stores its historical best-known ever position.

Step 2. Compute the evaluation value (V) for the population, store the one with best evaluation value particle as the g_{best}. Sort the population based on the evaluation value in a ascending order.

Step 3. Take a particle into ***Updating mode*** (Introduced in next section), and update its searching location. Do the same updating process for all the particles.

Step 4. Evaluate the new locations of particles, and compare the current evaluation value with its p_h value. If the current location is better then replace the p_h value.

Step 5. Loop to *step* 2 until a condition is met, either the loop index reaches a maximum value or the g_{best} is satisfied with the need.

2.2 Updating Mode

Suppose particle j, P_j is under updating mode, its $X_j = \{x_{j,1}, x_{j,2}, \ldots, x_{j,D}\}$ and $V_j = \{v_{j,1}, v_{j,2}, \ldots, v_{j,D}\}$. Copy $\widetilde{p}_j = p_j$.

Step 1. Set $MC = MC + 1$, if MC is smaller than $threshold_1$, change the velocity and position of $\widetilde{p}_j$ based on Eqs. 1 and 2. Otherwise, exit the updating mode. in the equation w, c_1 and c_2 are positive constants, r_1 and r_2 are two random numbers in the range [0,1], symbol $i \in [1, 2, \ldots, D]$.

$$v_{j,i}(t+1) = w \times v(t) + c_1 \times r_1 \times (x_{p_h,i} - x_{j,i}) + c_2 \times r_2 \times (x_{g_{best},i} - x_{j,i}) \tag{1}$$

$$x_{j,i}(t+1) = x_{j,i}(t) + v_{j,i}(t+1) \tag{2}$$

Step 2. Compare the $\widetilde{p}_j$ to the p_j, if $\widetilde{p}_j$ is better, then replace p_j with $\widetilde{p}_j$ and go back to *Step* 1. Otherwise, check the MC value, if MC equals to 0, go to *step* 3, If not, exit the updating mode.

Step 3. Generate a random number *ran*. if *ran* is smaller than $threshold_2$. Use Eq. 3 to update p_j. Otherwise, use Eq. 4. In the Eq. 3 m is a integer, which controls the number of good particles $gp = \{gp_1, gp_2, \ldots, gp_m\}$ used in advantage sharing process. In the Eq. 4, $x_{boundary}$ is a searching space related value.

$$\begin{gathered} gp_i = r + j, \quad r \in [0, N - j] \\ x_{j,i} = (\textstyle\sum_{k=1}^{m} x_{gp_k,i}) \div m \end{gathered} \tag{3}$$

$$x_{j,i}(t+1) = rand \times x_{boundary}, \quad rand \in [0, 1] \tag{4}$$

2.3 Pseudocode of AS-PSO

The pseudocode of AS-PSO is showed as

3 Experiment Results

Three different algorithms are simulated in our experiment: *PSO*, *SLPSO*, our scheme. Rosenbrock (5), Rastrigrin (6) and Sphere (7) this three benchmark functions are used to test their performance. The benchmark function $f(x)$ is shown as follows:

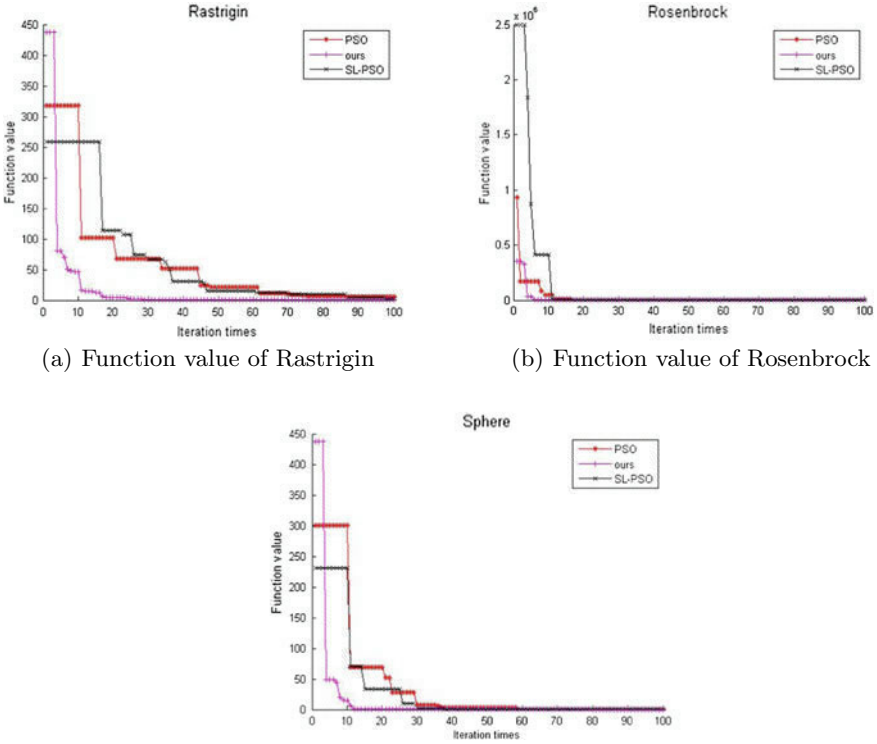
Algorithm 1 Pseudocode of AS-PSO

function AS- PSO($P = \{p_1, p_2, \ldots, p_N\}, g_{best}$)
Input $P, threshold_1, threshold_2, D, c_1, c_2, x_{boundary}$
Output: g_{best}
  for $i = 0$ to N **do**
    Initialize p_i, Compute V(p_i)
    Store p_h
  end for
  Store g_{best}
  for $t = 1$ to iteration times **do**
    for $j = 0$ to N **do**
      $\widetilde{p_j} = p_j$
      $MC = 0$
      for MC to $threshold_1$, $MC+ = 1$ **do**
        for $i = 0$ to D, do $\widetilde{p_j}$ **do**
          $v_{j,i}(t+1) = v(t) + c_1 \times r_1 \times (x_{p_h,i} - x_{j,i}) + c_2 \times r_2 \times (x_{g_{best},i} - x_{j,i})$
          $x_{j,i}(t+1) = x_{j,i}(t) + v_{j,i}(t+1)$
        end for
        if V(p_j)is worse than V($\widetilde{p_j}$) **then**
          $p_j = \widetilde{p_j}$
        end if
      end for
      if $ran \leq threshold_2$,do p_j **then**
        Sort(P)
        if $MC = 0$ **then**
          for $i = 0$ to m **do**
            $gp_i = r + j, \quad r \in [0, N - j]$
            $x_{j,i} + = (\sum_{k=1}^{m} x_{gp_k,i}) \div m$
          end for
        else
          for $i = 0$ to m **do**
            $x_{j,i}(t+1) = rand \times x_{boundary}, \quad rand \in [0, 1]$
          end for
        end if
      end if
    end for
    Store p_h for P, Store g_{best}
  end for
  Output g_{best}
end function

$$f_1(x) = f(x_1, x_2, \ldots, x_d) = \sum_{i=1}^{d} [100 \times (x_{2i-1}^2 - x_{2i})^2 + (x_{2i} - 1)^2] \tag{5}$$

Table 1 Symbols description

Symbols	Description	Values
N	The number of population	16
D	The number of dimension	3
m	The good particles used in advantage sharing	2
c_1, c_2	Acculate coefficent	1.49445
$x_{boundary}$	The boundary value of space	$[-100, 100]$
$threshold_1$	The steps value in Updating model process	2
$threshold_2$	The probability of random walk and advantage sharing	0.5
iteration	The iteration times used in three algorithm	100



(a) Function value of Rastrigin

(b) Function value of Rosenbrock

(c) Function value of Sphere

Fig. 1 Function value convergence curves

$$f_2(x) = f(x_1, x_2, \ldots, x_d) = Ad + \sum_{i=1}^{d} \left[x_i^2 - A \times \cos(2\pi x_i)\right] \tag{6}$$

Table 2 Comparison results in Function value

BenchMark	Methods		
	PSO	SLPSO	AS-PSO
$f_1(x)$	4248.89485	0.53217	0.20247
$f_2(x)$	6.35623	2.55794	0.000000
$f_3(x)$	0.13459	0.00036	0.000000

Table 3 The g_{best} positions in Rastrigrin Function

Values	Methods		
	PSO	SLPSO	AS-PSO
x_1	1.057379	−0.065119	7.132366×10^{-8}
x_2	−0.0499616	0.0495942	-3.333312×10^{-8}
x_3	2.014951	0.080029	2.703721×10^{-8}

$$f_3(x) = f(x_1, x_2, \ldots, x_d) = \sum_{i=1}^{d} x_i^2 \tag{7}$$

The parameters used in experiment are showed in Table 1.

The convergence curves of three benchmark function in three algorithms are showed in Fig. 1, the x-coordinate is the iteration times, and the y-coordinate is the function value. The AS-PSO has the best convergence speed, followed by SLPSO, and finally PSO.

Table 2 lists the final benchmark function value, from the table we can tell that *AS-PSO* shows a better performance, the benchmark function value are around 0.000000, which is close to the minimum value of Rastrigin and Sphere. Table 3 lists the final position x_1, x_2, x_3 of g_{best} in Rastrigin function, its position axes should be (0.0, 0.0, 0.0) in minimum value. From the table, it tells AS-PSO is better than the other comparison algorithms.

4 Conclusion

In this paper, we propose a variant of particle swarm optimization with the idea of advantage sharing and pre-learning walk. One particle evolves its position based on several other excellent individuals instead of one best-known member, which is Advantage sharing. It is right in a big probability to go more steps along the benefit obtained way. Furthermore, this algorithm also adds a small proportional random walk operation for avoiding converging local optimum. In the end, the original PSO and a recently proposed improved PSO are simulated. The experimental results show that AS-PSO has a good convergence speed and the final results are better than the comparison algorithms.

References

1. Derrac, J., Salvador, G., Daniel, M., Francisco, H.: A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. Swarm Evolut. Comput. **1**(1), 3–18 (2011)
2. Shu-Chuan, C., Pei-Wei, T., Jeng-Shyang, P.: Cat swarm optimization. In: Pacific Rim International Conference on Artificial Intelligence. Springer, Berlin, Heidelberg (2006)
3. Thi-Kien, D., Tien-Szu, P., Trong-The, N., Shu-Chuan, C.: A compact articial bee colony optimization for topology control scheme in wireless sensor networks. J. Inf. Hiding Multimed. Signal Process. **6**(2), 297–310 (2015)
4. Wuling, R., Cuiwen, Z.: A localization algorithm based On SFLA and PSO for wireless sensor network. Inf. Technol. J. **12**(3), 502–505 (2013)
5. Kaur, P., Shikha, M.: Resource provisioning and work flow scheduling in clouds using augmented Shuffled Frog Leaping Algorithm. J. Parallel Distrib. Comput. **101**, 41–50 (2017)
6. Ishaque, K., Zainal, S., Muhammad, A., Saad, M.: An improved particle swarm optimization (PSO) based MPPT for PV with reduced steady-state oscillation. IEEE Trans. Power Electron. **27**(8), 3627–3638 (2012)
7. Kennedy, J.: Particle swarm optimization. In: Encyclopedia of machine learning, pp. 760–766. Springer, Boston, MA (2011)
8. Marinakis, Y., Magdalene, M.: A hybrid genetic particle swarm optimization algorithm for the vehicle routing problem. Expert Syst. Appl. **37**(2), 1446–1455 (2010)
9. Nickabadi, A., Mohammad, M.E., Reza, S.: A novel particle swarm optimization algorithm with adaptive inertia weight. Appl. Soft Comput. **11**(4), 3658–3670 (2011)
10. Moradi, M.H., Abedini, M.: A combination of genetic algorithm and particle swarm optimization for optimal DG location and sizing in distribution systems. Int. J. Electrical Power Energy Syst. **34**(1), 66–74 (2012)
11. Niknam, T.: A new fuzzy adaptive hybrid particle swarm optimization algorithm for non-linear, non-smooth and non-convex economic dispatch problem. Appl. Energy **87**(1), 327–339 (2010)
12. Cheng, R., Yaochu, J.: A social learning particle swarm optimization algorithm for scalable optimization. Inf. Sci. **291**, 43–60 (2015)

Density Peak Clustering Based on Firefly Algorithm

Jiayuan Wang, Tanghuai Fan, Zhifeng Xie, Xi Zhang and Jia Zhao

Abstract In density peak clustering the choice of cut-off distance is not theoretically supported, and to address this concern, we propose density clustering based on firefly algorithm. The certainty between data is determined on the basis of density estimation entropy. The cut-off distance corresponding to the minimum entropy is found by iterative optimization of FA, and then substituted into the standard density clustering algorithm. Simulation experiments are conducted on eight artificial datasets. Compared with the standard density peak clustering, our method can choose the cut-off distance in a self-adaptive manner on different datasets, which improves the clustering effect.

Keywords Density peak clustering · Cut-off distance · Density estimation entropy · Firefly algorithm

1 Introduction

Clustering is a process of clustering dataset samples into clusters on the basis of similarity. As a result, samples within the same cluster have a higher similarity, and those across different clusters are low in similarity [1–3]. Clustering is able to extract

J. Wang · T. Fan (✉) · Z. Xie · X. Zhang · J. Zhao
School of Information Engineering, Nanchang Institute of Technology, Nanchang 330099, China
e-mail: fantanghuai@163.com

J. Wang
e-mail: wangjiayuan1201@163.com

Z. Xie
e-mail: xiezhifeng_nit@163.com

X. Zhang
e-mail: zhangxi95@vip.qq.com

J. Zhao
e-mail: zhaojia925@163.com

J.-S. Pan et al. (eds.), *Genetic and Evolutionary Computing*,
Advances in Intelligent Systems and Computing 834,
https://doi.org/10.1007/978-981-13-5841-8_2

hidden pattern and rules from a large amount of data and considered an important means to identify knowledge from the data. At present, clustering has found extensive applications in data analysis [4] and engineering system [5].

Many clustering algorithms have emerged to deal with different types of datasets. Conventional clustering algorithms include partitioning-based clustering [6], hierarchical clustering method [7], density-based clustering method [8] and network-based clustering method [9]. To improve the clustering efficiency and reduce complexity, Rodriguez et al. [10] described clustering by fast search and find of density peaks (DPC) in 2014. This algorithm can rapidly find the density peak points for datasets of any shape and efficiently perform the assignment of data points and removal of outliers. Therefore, DPC is fit for clustering analysis over massive data. DPC is based on the concept of cut-off distance d_c, used as a density measure for the samples. The choice of d_c has a large impact on the clustering effect. However, d_c is generally determined subjectively with the current DPC, and to do this, the empirical value first needs to be obtained from extensive prior experiments for different datasets, which is definitely a major drawback for DPC.

Firefly Algorithm (FA) [11–14] is a new optimization technique based on swarm intelligence [15], proposed by Yang Xin-she in 2008. In this algorithm, each firefly is randomly distributed in the solution space and has a fitness value assigned by optimization corresponding to the intensity of the emitted light. The firefly determines the direction of its movement by comparing the intensity of the emitted light, and the distance of movement is determined by relative attractiveness. After the population evolves for some generations, most fireflies will be attracted to the firefly with the highest intensity of the emitted light, and the optimization task is completed. FA is structurally simple, needs fewer parameters and applies to many optimization problems.

Here, FA is introduced to address the drawbacks of DPC, and density peak clustering based on firefly algorithm (FADPA) is proposed as a modified approach. This novel algorithm uses density estimation entropy as the objective function of d_c, which is optimized iteratively by FA. Cut-off distance d_c is determined self-adaptively for different datasets, and then the clustering is performed by using standard DPC.

2 Relevant Studies

2.1 Density Peak Clustering

DPC can automatically identify the cluster center of the dataset samples and achieve high-efficiency clustering for datasets of any shape. Its basic working principle is as follows. An ideal clustering center has two basic features: (1) the local density of the cluster center is higher than that of its neighbors; (2) the distance between different cluster centers is relatively large. To find the cluster centers that meet the above criteria, the local density ρ_i for sample i and the distance δ_i from sample i

to sample j that has a local density greater than sample i and is nearest to it. It is defined as follows:

$$\rho_i = \sum_{j \neq i} \chi(d_{ij} - d_c), \tag{1}$$

where d_{ij} is the Euclidean distance between sample i and sample j; d_c is the cut-off distance. When $x < 0$, $\chi(x) = 1$; otherwise $\chi(x) = 0$.

$$\delta_i = \min_{j:\rho_j > \rho_i} (d_{ij}) \tag{2}$$

For sample i with the largest local density ρ_i, $\delta_i = \max(d_{ij})$.

It is known from formula (1) that the local density ρ of the sample is a discrete value and greatly influenced by the cut-off distance d_c. To reduce the influence of d_c on the local density ρ of the sample, literature [16] uses a Gaussian kernel function to compute the local density of the sample.

$$\rho_i = \sum_{j \neq i} \exp\left(-\left(\frac{d_{ij}}{d_c}\right)^2\right) \tag{3}$$

DPC constructs the decision graph of relative distance δ about local density ρ. Samples with large δ and ρ are the density peak points. However, when the size of each cluster is small, the difference between ρ and δ is insignificant for the sample. As a result, the samples are sparse and the density peak points are unclear. So it is very difficult to find the density peak points by comparing ρ or δ alone. To solve this problem, ρ and δ are normalized [17]. The cluster centers are found by using γ decision graph, and are defined as follows:

$$\gamma_i = \rho_i \cdot \delta_i \tag{4}$$

Apparently, sample points with large γ can be chosen as the centers. For the remaining sample j, DPC categorizes it into the cluster where the sample with a larger local density than j and nearest to j is located. In this way, the remaining sample is assigned.

2.2 *Firefly Algorithm*

FA is an optimization process based on swarm intelligence, which mimics the behavior of fireflies in nature attracting the opposite gender by the emitted light. To reduce the algorithm complexity, the gender of the fireflies is neglected. It is generally believed that the fireflies emitting less bright light will be attracted to those emitting

brighter light. The movement distance is determined by the relative attractiveness between the fireflies. In the standard FA, relative attractiveness between the fireflies is given by

$$\beta = \beta_0 e^{-\gamma r_{ij}^2}, \tag{5}$$

where r_{ij} is the Euclidean distance between firefly i and firefly j; β_0 is the attractiveness of the firefly at the distance $r_{ij} = 0$, a constant; γ is the light absorption parameter, usually set to 1. For any two fireflies, their Euclidean distance is given by

$$r_{ij} = \left\| x_i - x_j \right\| = \sqrt{\sum_{d=1}^{D} \left(x_{id} - x_{jd} \right)^2}, \tag{6}$$

where D is the problem dimension.

Hence the movement of firefly i to firefly j can be defined by

$$x_{id}(t+1) = x_{id}(t) + \beta_0 e^{-\gamma r_{ij}^2} (x_{jd}(t) - x_{id}(t)) + \alpha(t)\varepsilon, \tag{7}$$

where x_{id} and x_{jd} are the d-th dimension of firefly i and j, respectively. Parameter $\varepsilon = (rand() - 1/2)$, and $rand()$ is a random function distributed uniformly in the range [0, 1]. $\alpha(t)$ is the step factor, with the value range [0, 1]. t is the number of iterations.

Through the above, the position of firefly is translated into an optimization problem. The brightness of the firefly is the function value of the optimization problem. The position of the firefly is constantly updated through iterative optimization of the firefly population, until the optimal solution is found to the solution.

3 Density Peak Clustering Based on Firefly Algorithm (FADPC)

DPC defines different d_c for different datasets and achieves a good clustering effect. However, the choice of d_c is not supported by theory, and is usually done according to the general principle of d_c ensuring that the mean number of neghbours for each data point accounts for about 1–2% of the total data points. In this study, the number of neiighbors is the number of data points with $d_{ij} < d_c$. This principle is only an empirical one drawn from several datasets, and its universality remains to be verified.

FADPC aims for a more reasonably selected d_c by constructing an objective function to solve d_c by FA. Thus appropriate d_c values are obtained self-adaptively for different datasets, and the clustering is made more accurate.

3.1 Density Estimation Entropy

By the definition of DPC, the sample points are ranked in accordance with the local density ρ. The δ values are computed sequentially, and the samples with high δ and ρ are identified as clusters. Moreover, the samples are ranked and categorized in a decreasing order of ρ. If any sample is miscategorized, the samples following it will be also miscategorized. Therefore, the objective function is designed to make all ρ values of the samples to be uniformly distributed in a decreasing order as much as possible.

Information entropy [18] is a measure of system uncertainty and has found applications in clustering algorithm in recent years. Consider a dataset $D = \{x_1, x_2, \ldots, x_n\}$ containing n samples in space. Let the value of density function of each sample be $\varphi_i = \sum_{j=1}^{n} e^{\left(\frac{\|x_i - x_j\|}{\delta}\right)^2}$, then the density estimation entropy is defined as

$$H = -\sum_{i=1}^{n} \frac{\varphi_i}{Z} \log\left(\frac{\varphi_i}{Z}\right), \tag{8}$$

where $Z = \sum_{i=1}^{n} \varphi_i$ is the normalization factor.

By comparing formula (8) and formula (3), it can be found that δ in the value of density function for each sample has the same meaning as the cut-off distance d_c. Optimizing δ is in essence to optimize the cut-off distance d_c.If the entire dataset is considered as a system, the best clustering effect can be achieved when the entire system is most stable and the relationship between the data has the highest certainty.

To obtain δ corresponding to the minimum density estimation entropy H, optimization is performed by FA. The density estimation entropy H is taken as the objective function to be solved. Each firefly particle represents a δ value, and FA is implemented to obtain a δ that makes H value minimum for the clustering. This approach can overcome the drawback of manual parameter configuration and improve the accuracy of clustering.

3.2 Steps of FADPC

Formula (8) is used as the objective function for FA, which is then implemented to optimize δ. The optimized result is substituted as d_c into DPC for the clustering. The implementation steps of the algorithm are shown below

Step 1: Choose dataset sample $A_{m\times n}$ (m is the number of data points, and n is the dimension). Calculate the distance matrix $D_{m\times m}$ (Euclidean distance between any two data points) for the dataset.