A novel pigeon-inspired optimization with QUasi-Affine TRansformation evolutionary algorithm for DV-Hop in wireless sensor networks

Xiao-Xue Sun¹, Jeng-Shyang Pan¹, Shu-Chuan Chu¹,², Pei Hu¹,³ and Ai-Qing Tian¹

Abstract
In modern times, swarm intelligence has played an increasingly important role in finding an optimal solution within a search range. This study comes up with a novel solution algorithm named QUasi-Affine TRansformation-Pigeon-Inspired Optimization Algorithm, which uses an evolutionary matrix in QUasi-Affine TRansformation Evolutionary Algorithm for the Pigeon-Inspired Optimization Algorithm that was designed using the homing behavior of pigeon. We abstract the pigeons into particles of no quality and improve the learning strategy of the particles. Having different update strategies, the particles get more scientific movement and space exploration on account of adopting the matrix of the QUasi-Affine TRansformation Evolutionary algorithm. It increases the versatility of the Pigeon-Inspired Optimization algorithm and makes the Pigeon-Inspired Optimization less simple. This new algorithm effectively improves the shortcoming that is liable to fall into local optimum. Under a number of benchmark functions, our algorithm exhibits good optimization performance. In wireless sensor networks, there are still some problems that need to be optimized, for example, the error of node positioning can be further reduced. Hence, we attempt to apply the proposed optimization algorithm in terms of positioning, that is, integrating the QUasi-Affine TRansformation-Pigeon-Inspired Optimization algorithm into the Distance Vector–Hop algorithm. Simultaneously, the algorithm verifies its optimization ability by node location. According to the experimental results, they demonstrate that it is more outstanding than the Pigeon-Inspired Optimization algorithm, the QUasi-Affine TRansformation Evolutionary algorithm, and particle swarm optimization algorithm. Furthermore, this algorithm shows up minor errors and embodies a much more accurate location.

Keywords
Pigeon-inspired optimization algorithm, QUasi-Affine TRansformation evolutionary algorithm, evolution matrix, DV-Hop algorithm

Date received: 10 January 2020; accepted: 1 May 2020

Handling Editor: Kuo-Ming Chao

Introduction
The optimization problem is the values of a set of parameters under certain constraint conditions, so as to obtain a certain performance, and play a role to a certain extent. It usually solves mathematical problems. Swarm intelligence algorithms are mostly stemmed

¹College of Computer Science and Engineering, Shandong University of Science and Technology, Qingdao, China
²College of Science and Engineering, Flinders University, Adelaide, SA, Australia
³School of Software, Nanyang Institute of Technology, Nanyang, China

Corresponding author:
Jeng-Shyang Pan, College of Computer Science and Engineering, Shandong University of Science and Technology, Huangdao, Qingdao 266590, China. Email: jspan@cc.kuas.edu.tw

Creative Commons CC BY. This article is distributed under the terms of the Creative Commons Attribution 4.0 License (https://creativecommons.org/licenses/by/4.0/) which permits any use, reproduction and distribution of the work without further permission provided the original work is attributed as specified on the SAGE and Open Access pages (https://us.sagepub.com/en-us/nam/open-access-at-sage).
from biological behaviors in nature, and most of them have strong optimization ability, which is a powerful tool to solve problems in life. In the last few years, a good deal of naturally inspired computation methods are proposed, which have their own expansibilities and have been widely used in engineering and daily life. Although the Pigeon-Inspired Optimization (PIO) algorithm has obvious advantages such as faster convergence speed and simpler calculation comparing with other algorithms, it still needs further study in the basic theory of convergence and multi-objective optimization.

In the early years, Duan and Qiu propose pigeon-inspired optimization which comes out the simulation of homing behavior of pigeons through magnetic field and landmarks. Pigeons could seek out their homing tools with no difficulty: magnetic fields, the sun, and landmarks. The homing ability of pigeons has been known, and pigeons are used as communication tools a long time ago. When the pigeons are far away from their destination, they use the geomagnetic field to identify the direction, and when they are closer to the destination they use the local landmark to navigate. The first model in PIO algorithm is based on the geomagnetic field and the sun. While the landmark operator model is based on landmarks.

The PIO algorithm has good performance, but it has certain limitations. Based on the simple structure of the PIO algorithm and the peculiarity of simply combining with other intelligent algorithms, we bring forward an improved algorithm that combined with the QUasi-Affine TRansformation Evolutionary (QUATRE) algorithm. QUATRE is a new metaheuristic algorithm proposed by Meng in 2016, which uses an evolutionary formula like the affine transformation in geometry. The PIO makes use of a group of pigeons to delegate candidate solutions in the solution space to be studied and the optimization problem is that pigeons remove toward the best solution for a given measurement of performance through iteration.

In this article, our proposed algorithm combines the PIO algorithm with the QUATRE algorithm. This novel QUasi-Affine TRansformation-Pigeon-Inspired Optimization (QT-PIO) evolutionary optimization algorithm still comprises two operators. In the primary stage, the map and compass operator is executed to explore and exploit the solution space for multiple iteration. The particles learning strategy combines evolutionary guidance matrix B and co-evolutionary matrix M of QUasi-Affine TRansformation Evolutionary Algorithm into matrix form. In the second phase, the population is divided into two groups. In the first part, the population size reduces from ps to ps/2. The second part of particles start their new explorations.

With the rapid development of information science and technology, the requirement of node location is becoming more and more accurate in wireless sensor networks (WSNs). Nowadays, WSNs are widely applied in various fields, for example, intelligent transportation, defense, environmental monitoring, health care, space exploration, and many other fields. The Distance Vector–Hop (DV-Hop) positioning algorithm is one of a series of distributed localization methods on the strength of distance-vector routing and Global Positioning System (GPS) localization.

The DV-Hop node localization algorithm is an important distance-independent positioning algorithm. DV-Hop is a node localization algorithm that is independent of signal attenuation, and it has high practicability in terms of network cost, layout, and signal attenuation. In a randomly distributed network, the node positioning error is large. More and more intelligence algorithms are used to optimize DV-Hop, this paper mainly takes advantage of QT-PIO to optimize that decreases the error of positioning, makes the location more accurate, and thus ensures a certain degree of accuracy.

This text mainly proposes a new approach with regard to finding an optimal solution. The optimization ability performs well and effectively improves the convergence speed in this model. Applying it to the DV-Hop algorithm and the experimental results indicate that our consideration greatly reduces errors of positioning and improves positioning accuracy. In a word, it is justifiable to select QT-PIO to find the optimum solution. This article is separated into five sections, the second section describes the basis of the new algorithm. The third section raises our novel algorithm. The fourth section describes the consequence of evolutionary optimization. Next, the fifth section discusses the optimal DV-Hop algorithm. The last section is conclusion.

**Related work**

This section mainly exhibits the basis of the proposed algorithm, including the original PIO algorithm and the QUATRE algorithm.

**Canonical PIO algorithm**

On the strength of the specific navigational behavior of the pigeons in their search for a home, a bionic group intelligent optimization algorithm is proposed. PIO algorithm puts to use two diverse operator model two disparate that is put forward by means of simulating the mechanism in which the pigeons make use of various guide instruments in a disparate stage to find the
loft. There are diverse tools: the map and compass operator and landmark operator. The pigeons perceive the field of the earth with magnetic material as their map and think of the height of the sun as a compass.

In the loft model, the objects of the study is the particle which is virtualized from pigeons in the navigation process.\textsuperscript{10,11} Initial setting of the parameters are set, for example, the population of pigeons is \( ps \), the pigeons’ search in \( D \)-dimensional space, the number of running is set to 100, the current iteration is \( \text{iter} \), the total number of iterations is \( \text{Iter}_{\text{max}} = 600 \), in the first stage iterations are \( \text{maxIteration}_1 = 300 \), in the second phase iterations are \( \text{maxIteration}_2 = 300 \), and the position and velocity of every pigeon are recorded as below (\( j = 1, 2, 3, ..., \text{ps} \))

\[
\vec{x}_j = \left[ x_{j,1}, x_{j,2}, x_{j,3}, ..., x_{j,D} \right], \tag{1}
\]

\[
\vec{v}_j = \left[ v_{j,1}, v_{j,2}, v_{j,3}, ..., v_{j,D} \right], \tag{2}
\]

where \( j \) means the \( j \)th individual in the population, \( \vec{x}_j \) represents the coordinates of \( j \)th individual and \( \vec{v}_j \) signifies the \( j \)th velocity.

When \( \text{iter} \) is greater than 1 and less than \( \text{maxIteration}_1 \), the map and compass operator has an effect on the homing of the pigeons. The pigeon \( X_i \) updates its velocity at \((\text{iter} + 1)^{\text{th}}\) by

\[
\vec{v}_j^{\text{iter} + 1} = e^{-R \text{iter}(\text{iter} + 1)} \ast \vec{v}_j^{\text{iter}} + b \ast (\vec{X}_{\text{best}}^{\text{iter}} - \vec{x}_j^{\text{iter}}), \tag{3}
\]

where \( R \) is a positive number that serves as the first factor, \( b \) is a numerical value randomly generated between 0 and 1, and \( \vec{X}_{\text{best}}^{\text{iter}} \) is the global optimal position obtained by making a comparison among the positions of all the pigeons after \( \text{iter} \) iterative loop.

Affected by the map and compass operator, pigeon adjusts its position with the new velocity of each pigeon at \((\text{iter} + 1)^{\text{th}}\) as below

\[
\vec{x}_j^{\text{iter} + 1} = \vec{x}_j^{\text{iter}} + \vec{v}_j^{\text{iter} + 1}. \tag{4}
\]

When \( \text{iter} \) is greater than the set maximum in the first phase, the exploration and exploitation are stopped immediately, and the optimization proceeds into the following phase which takes advantage of the landmark operator. In this process, half of the pigeons is deserted with the passage of each iteration, which navigate toward the loft when the number of iterations \( \text{iter} \) is from \( \text{maxIteration}_1 \) to \( \text{maxIteration}_1 + \text{maxIteration}_2 \). Pigeons that are far from their destination are considered to have no ability to distinguish their home path, so they are discarded. While the current number of iterations is less than \( \text{maxIteration}_1 + \text{maxIteration}_2 \), the position \( \vec{x}_j \) of each remaining pigeon is expressed as in the following expression

\[
\left\{
\begin{align*}
\vec{x}_j^{\text{iter} + 1} &= \vec{x}_j^{\text{iter}} + \vec{v}_j^{\text{iter} + 1}, & & \text{if } \text{iter} < \text{maxIteration}_1 + \text{maxIteration}_2, \\
\vec{x}_j^{\text{iter} + 1} &= \vec{x}_j^{\text{iter}} + \vec{v}_j^{\text{iter} + 1} + b \ast (\vec{X}_{\text{best}}^{\text{iter}} - \vec{x}_j^{\text{iter}}), & & \text{if } \text{iter} \geq \text{maxIteration}_1 + \text{maxIteration}_2.
\end{align*}
\right\
\]

\( \vec{x}_j^{\text{iter}} \) is the midpoint among the own pigeons left and serves as a landmark in the second phase, the reference direction in which the rest of the left pigeons will fly. And \( b \) is defined by a random value between 0 and 1. \( ps^{\text{iter}} \) is representative of the population of \( \text{iter} \) generation. For the different questions asked, \( F(\vec{x}_j^{\text{iter}}) \) can be solved as follows

\[
F(\vec{x}_j^{\text{iter}}) = f_{\text{obj}}(\vec{x}_j^{\text{iter}}),
\]

for the minimization problem and the maximization problem,

\[
F(\vec{x}_j^{\text{iter}}) = \frac{1}{f_{\text{obj}}(\vec{x}_j^{\text{iter}}) + a},
\]

where \( f_{\text{obj}}(\vec{x}_j^{\text{iter}}) \) is the function value of the pigeon \( \vec{x}_j \) at the \( \text{iter}^{\text{th}} \) generation and \( a \) is an any nonzero constant.\textsuperscript{12}

Similarly, the landmark operator stops working, supposing that \( \text{iter} \) exceeds the maximum iteration limit set previously at the present.

**QUATRE algorithm**

The evolutionary algorithm used in the QUATRE algorithm is similar to the affine transformation in geometry.\textsuperscript{13,14,15} The evolutionary architecture adopted by the QUATRE algorithm is \( X \rightarrow M \times X + \overline{M} \times B \), so the equation is given as below

\[
X = M \otimes X + \overline{M} \otimes B, \tag{8}
\]

where \( X \) stands for individual population matrix, \( X = (\vec{x}_1, \vec{x}_2, ..., \vec{x}_j, ..., \vec{x}_m)^{T}, \ j \in [1, ps] \). \( \otimes \) signifies the bitwise multiplication of matrix elements. Matrix \( B \) denotes the evolutionary mutation matrix and it has the same number of lines as Matrix \( M \). Matrix \( M \) is a co-evolutionary matrix, \( \overline{M} \) is an association matrix \( M \), and \( \overline{M} \) represents a binary inversion operation of the elements of \( M \). Elements from the same position of the two matrices have opposite values, for example in equations (9) and (10).
The co-evolutionary matrix $M$ is received, after the matrix elementary transformation of $M_{\text{initial}}$. It represents the dimension of object function, takes individual population size $ps = D$ as an example. The matrix $M_{\text{initial}}$ is a lower triangular matrix with an element value of 1. The equation (11) gives an example of the transformation from the matrix $M_{\text{initial}}$ to the matrix $M$.

$$
\begin{bmatrix}
1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 \\
\end{bmatrix}
= 
\begin{bmatrix}
1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 \\
\end{bmatrix}
$$

Each individual $\hat{X}_{j} (j \in [1, ps])$ in the matrix $X$ has a corresponding evolutionary instruction vector $B$, which has several different generation methods given in Table 1. $X_{l,g}$ ($l \in [1, ps]$) means entirely the random matrix which is produced by arranging the line sequence of matrix $X$ of the $g$th generation at random. $\hat{X}_{\text{gbest},g}$ denotes the global best position at $g$th generation. The same $ps$ row vectors make up $X_{\text{gbest},g}$ and which $X_{\text{gbest},g}$ is shown in equation (13).
Commonly, the outcome of \((\tilde{X}_r_i - \tilde{X}_p_i)\) \((i, j \in [1, 2, ..., 5])\) is deemed to difference matrix. Commonly, \(F = 0.7\) is a good choice.

**QT-PIO algorithm**

The newly filed approach is an optimization algorithm combining pigeon-inspired optimization algorithm with a quasi-affine transformation algorithm. The basic PIO algorithm makes use of the first operator for exploration and takes advantage of the second operator for exploitation. The structure of the pigeon herd optimization algorithm is relatively simple. It combines with QUATRE algorithm, promoting and supplying each other, which enhances the capability of PIO to solve complex problems. The QT-PIO algorithm changes the iterative update strategy of particles and the particles still work in two consecutive phases to find the optimal solution. First, initialize the coordinate and speed of all particles in the form of a matrix using equations (14) and (15) which are given as follows

\[
\begin{bmatrix}
X_{\text{ghost}, g} \\
\tilde{X}_{\text{ghost}} \\
\tilde{X}_{\text{ghost}} \\
\cdot \\
\cdot \\
\cdot \\
\tilde{X}_{\text{ghost}}
\end{bmatrix}
\]  

\[
\begin{bmatrix}
\tilde{V}_1 \\
\tilde{V}_2 \\
\cdot \\
\cdot \\
\cdot \\
\tilde{V}_{ps}
\end{bmatrix}
\]

where \(\tilde{V}_j = [v_{j,1}, v_{j,2}, ..., v_{j,D}, ..., v_{ps,D}], j \in (1, 2, ..., ps)\)

\[\mathbf{B} = \mathbf{X}_{r_{1,g}} + F \ast (\mathbf{X}_{2,g} - \mathbf{X}_{3,g}), \]

\[\mathbf{V}_{iter + 1} = e^{-R \ast (iter + 1)} \ast \mathbf{V}_{iter} + b \ast (\mathbf{X}_{\text{ghost} - \mathbf{X}_{iter}}), \]

\[\mathbf{X}_{iter + 1} = \mathbf{M} \otimes \mathbf{X}_{iter} + \bar{\mathbf{M}} \otimes \mathbf{B} + r \ast \mathbf{V}_{iter + 1} . \]

In each iteration, under the mechanism of the first phase, all particles update their speed and position together in the form of a matrix, rather than updating one particle at a time. Ultimately, the velocity matrix and the position matrix are obtained after every iteration. After the position of species calculated by the evolutionary equation (16) are compared in pairs with individuals of the previous generation, the optimal individual is finally selected as the individual in the next generation population.

In the second optimization phase in QT-PIO, instead of abandoning the part of particles having relatively worse fitness values in the swarm \((ps = ps/2)\), this article considers that each particle has its own value. Whereas, there are two groups of particles and they adopt, respectively, disparate communication strategies to make them play different roles. At the first group, the population quantity of particles cut down linearly from \(ps\) to \(ps/2\) along with \(iter\) running and population

Table 1. The six mutation matrices \(\mathbf{B}\) calculation.

| Number | QUATRE/B | Equation |
|--------|----------|----------|
| 1      | QUATRE/target/1 | \(\mathbf{B} = \mathbf{X} + F \ast (\mathbf{X}_{1,g} - \mathbf{X}_{2,g})\) |
| 2      | QUATRE/rand/1   | \(\mathbf{B} = \mathbf{X}_{1,g} + F \ast (\mathbf{X}_{2,g} - \mathbf{X}_{3,g})\) |
| 3      | QUATRE/best/1   | \(\mathbf{B} = \mathbf{X}_{\text{ghost} } + F \ast (\mathbf{X}_{2,g} - \mathbf{X}_{3,g})\) |
| 4      | QUATRE/target/2 | \(\mathbf{B} = \mathbf{X} + F \ast (\mathbf{X}_{1,g} - \mathbf{X}_{2,g}) + F \ast (\mathbf{X}_{3,g} - \mathbf{X}_{4,g})\) |
| 5      | QUATRE/rand/2   | \(\mathbf{B} = \mathbf{X}_{1,g} + F \ast (\mathbf{X}_{2,g} - \mathbf{X}_{3,g}) + F \ast (\mathbf{X}_{4,g} - \mathbf{X}_{5,g})\) |
| 6      | QUATRE/best/2   | \(\mathbf{B} = \mathbf{X}_{\text{ghost} } + F \ast (\mathbf{X}_{1,g} - \mathbf{X}_{2,g}) + F \ast (\mathbf{X}_{3,g} - \mathbf{X}_{4,g})\) |

\[\mathbf{B}_{\text{QUATRE}}: \text{QUasi-Affine TRansformation Evolutionary.}\]
quantity after changing in the first group is recorded as \( ps1 \). Without a doubt, the spare particles are the second group. The first set of particles still uses the center position \( \mathbf{X}_{iter} \) of the remaining particles as the landmark and that acts as their exploitation direction to the best final solution. The exact evolution equation described in equation (17)

\[
\begin{align*}
ps1 & = \left[ \frac{(-C)}{\text{maxIteration}_2} \right] \times \text{iter} + ps1, \\
\hat{X}_{iter} & = \frac{1}{ps1 \times \sum_{iter} F(\hat{X}_{iter})}, \\
\tilde{X}_{iter} + 1 & = \tilde{X}_{iter} + b * (\tilde{X}_{iter} - \tilde{X}_{iter}),
\end{align*}
\]

where \( \text{maxIteration}_2 \) is the amount of iterations at the second phase, and \( F(\hat{X}_{iter}) \) is the weight of the \( f^6 \)th particle, which is computed by equation (18) on the basis of the maximization or minimization of the requested questions

\[
F(\tilde{X}_{iter}) = \begin{cases} 
\text{fobj}(\tilde{X}_{iter}), & \text{for minimization} \\
\frac{1}{\text{fobj}(\tilde{X}_{iter}) + a}, & \text{for maximization}
\end{cases}
\]

That is all shown in Algorithm 1.

**Algorithm 1. Pseudo Code of QT-PIO algorithm.**

1: Input: Initialize \( D, \{R_{max}, R_{max}\}, ps, F, R, a, \text{iter}, \text{maxIteration}_1, \text{maxIteration}_2, V, X, f(\tilde{X}) \);
2: //Compute the fitness value of every particle
3: for \( i = 1 : ps \) do
4: initialization of evaluation fitness value \( f(X_{iter}) \);
5: end for
6: Set the previous position \( X_{best} = X_{iter} \), optimal solution location \( X_{best} \), the optimal value \( f(X_{best}) \);
7: Generate the co-evolution matrix \( M \) according to equation (11)
8: for \( \text{iter} = 1 : \text{maxIteration}_1 \) do
9: \( B = X_{iter} + F \times (X_{iter} - X_{best}) \);
10: // Update position and velocity\( // \)
11: \( X_{iter} = M \times X_{iter} + M \times X_{best} + b * (X_{iter} - X_{iter}) \);
12: //Best Selection\( // \)
13: if \( f(X_{iter}) < f(X_{best}) \) then
14: \( X_{best} = X_{iter} \);
15: end if
16: Sort the particles and arrange them from smallest to largest\( // \)
17: The number of particles decreases linearly from \( ps \) to \( \frac{ps}{2} \) with the iterations that recorded as \( ps1 \) //
18: Select the center position from \( ps1 \) historical best points:
19: \( X_{best} = \frac{1}{ps1 \times \sum_{iter} F(\hat{X}_{iter})} \times \sum_{iter} (\hat{X}_{iter} - \hat{X}_{iter}) \);
20: // Update position and velocity\( // \)
21: for \( i = 1 : ps1 \) do
22: \( \tilde{X}_{iter} = X_{iter} + b * (X_{iter} - \tilde{X}_{iter}) \);
23: end for
24: Select the center position from \( ps1 \) historical best points:
25: \( X_{best} = X_{iter} + F \times (X_{iter} - X_{best}) \);
26: for \( i = ps1 + 1 : ps \) do
27: \( X_{iter} = X_{iter} + b * (X_{iter} - \tilde{X}_{iter}) \);
28: end for
29: \( X_{best} = X_{iter} + b * (X_{iter} - \tilde{X}_{iter}) \);
30: //Best Selection\( // \)
31: if \( f(X_{iter}) < f(X_{best}) \) then
32: \( X_{best} = X_{iter} \);
33: end if
34: \( X = X_{best} \);
35: Mark the optimal value of the population \( f(X_{best}) \)
36: end for
37: Return \( f(X_{best}) \) and \( X_{best} \).

QTPIO: QUasi-Affine TRansformation-Pigeon-Inspired Optimization.
The novel algorithm is combined with PIO and QUATRE that introduces matrices and adopts new update strategies. In the second stage, the number of particles is linearly processed, but no one is discarded, so that enhances the vitality of the particles, and the multimorphy of the species is augmented, thereby improves competitiveness of this algorithm.

### Experimental results and analysis

On behalf of evaluating the newly proposed QT-PIO algorithm, twenty-four functions are selected to test the performance of the algorithm. We regard convergence speed and the minimum solution as the criteria of measurement in this article. There are twenty-four test functions applied, the dimension settings and search scope in the following tables. Table 2 shows the unimodal functions. Table 3 contains the multimodal functions. Table 4 displays fixed-dimension multimodal functions.

Everhart and Kennedy presented the particle swarm optimization (PSO) first in 1995, and its basic concept originated from the research on the foraging behavior of birds. PSO is originally inspired by the cluster activity of birds and a useful model is created by group intelligence. To validate the quality of the property

---

**Table 2. Unimodal functions in experiments.**

| Number | Function | Space          | Dimension | fmin |
|--------|----------|----------------|-----------|------|
| 1      | \(f_1(z) = \sum_{i=1}^{m} z_i^2\) | \([-100, 100]\) | 30        | 0    |
| 2      | \(f_2(z) = \sum_{i=1}^{m} |z_i| + \sum_{i=1}^{m} |z_i|\) | \([-10, 10]\) | 30        | 0    |
| 3      | \(f_3(z) = \sum_{i=1}^{m} \prod_{j=1}^{i-1} (z_j)\) | \([-100, 100]\) | 30        | 0    |
| 4      | \(f_4(z) = \max|z_i|, i \in [1, m]\) | \([-100, 100]\) | 30        | 0    |
| 5      | \(f_5(z) = \sum_{i=1}^{m} \left[100(z_{i+1} - z_i)^2 + (z_i - 1)^2\right]\) | \([-30, 30]\) | 30        | 0    |
| 6      | \(f_6(z) = \sum_{i=1}^{m} (|z_i + 0.5|)^2\) | \([-100, 100]\) | 30        | 0    |
| 7      | \(f_7(z) = \sum_{i=1}^{m} l + z_i^2 + \text{rand}(0, 1)\) | \([-1.28, 1.28]\) | 30        | 0    |

**Table 3. Multimodal functions in experiments.**

| Number | Function | Space          | Dimension | fmin |
|--------|----------|----------------|-----------|------|
| 8      | \(f_8(z) = \sum_{i=1}^{m} -z_i \cdot \sin(\sqrt{|z_i|})\) | \([-500, 500]\) | 30        | -12569 |
| 9      | \(f_9(z) = \sum_{i=1}^{m} z_i^2 - 10 \cdot \cos(2\pi z_i) + 10\) | \([-5.12, 5.12]\) | 30        | 0    |
| 10     | \(f_{10}(z) = -20 \cdot \exp\left(-0.2 \sqrt{\frac{1}{m} \sum_{i=1}^{m} z_i^2}\right)\) - \(\exp\left(-\frac{1}{m} \sum_{i=1}^{m} \cos(2\pi z_i) + 20 + 2.718\right)\) | \([-32, 32]\) | 30        | 0    |
| 11     | \(f_{11}(z) = \frac{1}{4000} + \prod_{i=1}^{m} z_i^2 - \prod_{i=1}^{m} \cos\left(\frac{z_i}{\sqrt{10}}\right) + 1\) | \([-600, 600]\) | 30        | 0    |
| 12     | \(f_{12}(z) = \frac{\pi}{m} \cdot \left\{10 \cdot \sin(\pi y_i) + \sum_{i=1}^{m-1} (y_i - 1)^2 \left[1 + 10 \cdot \sin^2(\pi y_i) + (y_m - 1)^2\right]\right\} + \sum_{i=1}^{m} u(z_i, 10, 100, 4),\) | \([-50, 50]\) | 30        | 0    |
|        | \(y_i = 1 + \frac{z_i - 1}{4} = u(z_i, a, k, m)\) = \(\left\{\begin{array}{ll} k(z_i - a), & z_i > a \\ 0, & a < z_i < a \\ k(z_i - a), & z_i > a \end{array}\right\}\) | \([-50, 50]\) | 30        | 0    |
| 13     | \(f_{13}(z) = 0.1 \cdot \left\{\sin^2(3\pi z_1) + \prod_{i=1}^{m-1} (z_i - 1)^2 \left[1 + \sin^2(3\pi z_i + 1)\right] + (z_m - 1)^2 \left[1 + \sin^2(2\pi z_m)\right]\right\} + \sum_{i=1}^{m} u(z_i, 10, 100, 4)\) | \([-50, 50]\) | 30        | 0    |
for the optimization problems of the QT-PIO algorithm compared with the other three algorithms: PIO algorithm, PSO algorithm, QUATRE algorithm. The parameters of the four algorithms used in the comparison process are expressed in Table 5. The data of the average and square deviation under the test functions of over 600 iterations are obtained, and the comparison is expressed in Table 6. In the values of each row, the best value is highlighted as shown. Observed from the comparison of Table 6, the new QT-PIO algorithm can often provide superior results in most selected test functions than other algorithms of over 100 runs.

QT-PIO algorithm wins twenty in twenty-four on mean value comparison as Table 6 shown. For the sake of more intuitively evaluating the quality of the QT-PIO algorithm, Figure 1 exhibits the curves of PIO, QUATRE, PSO, QT-PIO with the best final fitness value obtained minimum for the unimodal functions

Table 4. Fixed-dimension multimodal functions in experiments.

| Number | Function | Space | Dimension | fmin |
|--------|----------|-------|-----------|------|
| 14     | $f_{14}(z) = \left( \frac{1}{2^{25}} + \sum_{i=1}^{25} \frac{1}{2^{i-1}} \sum_{j=1}^{2^i-1} (z_j - a_j)^2 \right)^{-1}$ | $[-65, 65]$ | 2 | 1 |
| 15     | $f_{15}(z) = \sum_{i=1}^{25} \left[ a_i - \frac{z_i (a_i + b_i)}{b_i - x_i^2} \right]^2$ | $[-5, 5]$ | 4 | 0.00030 |
| 16     | $f_{16}(z) = 4z_1^2 - 2.1z_1^4 + \frac{1}{3}z_1^6$ | $[-5, 5]$ | 2 | -1.0316 |
| 17     | $f_{17}(z) = \left( z_1 - \frac{5.1}{4\pi^2} z_1^2 + \frac{5}{\pi} z_1 - 6 \right)^2$ | $[-5, 5]$ | 2 | 0.398 |
| 18     | $f_{18}(z) = \left( 1 + (z_1 + z_2 + 1)^2 \right) \cdot \left( 19 - 14z_1 + 3z_1^2 - 14z_2 + 6z_1z_2 + 3z_2^2 \right)$ | $[-2, 2]$ | 2 | 3 |
| 19     | $f_{19}(z) = \exp \left( -\sum_{i=1}^{10} z_i \right)$ | $[1, 3]$ | 3 | -3.86 |
| 20     | $f_{20}(z) = \sum_{i=1}^{5} \left[ (z_i - a_i)^2 + c_i \right]^{-1}$ | $[0, 1]$ | 6 | -3.32 |
| 21     | $f_{21}(z) = \sum_{i=1}^{5} \left[ (z_i - a_i)^2 + c_i \right]^{-1}$ | $[0, 10]$ | 4 | 10.5363 |
| 22     | $f_{22}(z) = \sum_{i=1}^{5} \left[ (z_i - a_i)^2 + c_i \right]^{-1}$ | $[0, 10]$ | 4 | 10.4028 |
| 23     | $f_{23}(z) = \sum_{i=1}^{5} \left[ (z_i - a_i)^2 + c_i \right]^{-1}$ | $[0, 10]$ | 4 | 10.5363 |
| 24     | $f_{24}(z) = \left( \sum_{i=1}^{5} (z_i^2 + z_{i+1}) \right)$ | $[-5.12, 5.12]$ | 2 | -186.7309 |

Table 5. Parameters settings.

| No. | Algorithm | Parameter values |
|-----|-----------|------------------|
| 1   | PIO       | $p_s = 60$, $R = 0.2$, $a = 0.1$, $V_{max} = 3$, $I_{iter\_max} = 600$ (maxIteration$_1 = 300$, maxIteration$_2 = 300$) |
| 2   | PSO       | $p_s = 60$, $d1 = 2$, $d2 = 2$, $w_{max} = 0.9$, $w_{min} = 0.2$, $V_{max} = 3$, $I_{iter\_max} = 600$ |
| 3   | QUATRE    | $p_s = 60$, $F = 0.7$, $V_{max} = 3$, $I_{iter\_max} = 600$ (maxIteration$_1 = 300$, maxIteration$_2 = 300$) |
| 4   | QT-PIO    | $p_s = 60$, $R = 0.2$, $a = 0.1$, $F = 0.7$, $V_{max} = 3$, $I_{iter\_max} = 600$ (maxIteration$_1 = 300$, maxIteration$_2 = 300$) |

PIO: pigeon-inspired optimization; PSO: particle swarm optimization; QUATRE: QUasi-Affine TRansformation Evolutionary; QT-PIO: QUasi-Affine TRansformation-Pigeon-Inspired Optimization.
## Table 6. Comparison results of mean value and standard deviation with diverse algorithm.

| Function | PIO | QUATRE | PSO | QTPIO |
|----------|-----|--------|-----|-------|
|          | Mean | Standard | Mean | Standard | Mean | Standard | Mean | Standard |
| f1       | 18.6332 | 3.3724 | 0.3529 | 0.1224 | \(5.858 \times 10^{-2}\) | \(2.674 \times 10^{-2}\) | \(1.2615 \times 10^{-8}\) | \(9.0690 \times 10^{-9}\) |
| f2       | 15.8862 | 2.6240 | 8.5978 | 2.2405 | \(4.6275 \times 10^{-2}\) | \(1.0710 \times 10^{-2}\) | \(4.3198 \times 10^{-2}\) | \(1.0648 \times 10^{-2}\) |
| f3       | 1.08e + 03 | 5.3629e + 02 | 2.2363e + 04 | 5.7981e + 03 | \(1.1402 \times 10^{-2}\) | \(4.1997 \times 10^{-2}\) | \(7.9563 \times 10^{-2}\) | \(2.5069 \times 10^{-2}\) |
| f4       | 1.0220 | 0.0355 | 0.8211 | 1.2727 | \(15.5464 \times 10^{-2}\) | \(2.7414 \times 10^{-2}\) | \(0.9950 \times 10^{-2}\) | \(0.0263 \times 10^{-2}\) |
| f5       | 7.1147e + 02 | 3.2299e + 02 | 4.2781e + 02 | 3.3989e + 03 | \(4.0640 \times 10^{-2}\) | \(3.8863 \times 10^{-2}\) | \(52.7145 \times 10^{-2}\) | \(52.1526 \times 10^{-2}\) |
| f6       | 42.3820 | 5.8131 | 0.3763 | 0.1549 | \(6.1402 \times 10^{-2}\) | \(2.6795 \times 10^{-2}\) | \(5.4458 \times 10^{-2}\) | \(2.1702 \times 10^{-2}\) |
| f7       | 41.9966 | 15.4374 | 12.5332 | 11.5875 | \(-2.6509 + 76\) | \(2.2743 \times 10^{-2}\) | \(11.2245 \times 10^{-2}\) | \(2.7448 \times 10^{-2}\) |
| f8       | -2.5491e + 02 | 4.3312e + 02 | -1.9841e + 02 | 1.6593e + 03 | \(1.4167 \times 10^{-2}\) | \(4.1905 \times 10^{-2}\) | \(1.6980 \times 10^{-2}\) | \(0.5486 \times 10^{-2}\) |
| f9       | 20.9807 | 4.1413 | 1.3267 + 02 | 13.1095 | \(20.9195 \times 10^{-2}\) | \(4.294 \times 10^{-2}\) | \(0.0178 \times 10^{-2}\) | \(0.0218 \times 10^{-2}\) |
| f10      | 3.0330 | 0.2944 | 20.1919 | 0.0311 | \(1.1674 \times 10^{-2}\) | \(0.0646 \times 10^{-2}\) | \(1.3498 \times 10^{-2}\) | \(0.3603 \times 10^{-2}\) |
| f11      | 0.7127 | 0.1705 | 0.0518 | 0.0466 | \(4.3788 \times 10^{-2}\) | \(0.2006 \times 10^{-2}\) | \(1.3498 \times 10^{-2}\) | \(0.2006 \times 10^{-2}\) |
| f12      | 3.9410 | 0.8346 | 0.3745 | 0.2006 | \(2.9840 \times 10^{-2}\) | \(4.5549 \times 10^{-2}\) | \(-1.8673 + 76\) | \(2.4756 \times 10^{-2}\) |
| f13      | 0.4578 | 0.2041 | 0.9082 | 0.3402 | \(-1.8667 \times 10^{-2}\) | \(0.1663 \times 10^{-2}\) | \(9.5477 \times 10^{-2}\) | \(3.3481 \times 10^{-2}\) |
| f14      | -1.1408e + 02 | 38.7487 | -1.8673e + 02 | 9.7939e -04 | \(-1.0316 \times 10^{-2}\) | \(1.5580 \times 10^{-2}\) | \(-1.0316 \times 10^{-2}\) | \(1.4504 \times 10^{-2}\) |
| f15      | 0.0302 | 0.0210 | 0.0011 | 0.3061e -05 | \(8.6872 \times 10^{-2}\) | \(0.0100 \times 10^{-2}\) | \(9.5477 \times 10^{-2}\) | \(3.3481 \times 10^{-2}\) |
| f16      | -1.0316 | 1.87e -08 | -1.0316 | 1.4605e -15 | \(1.5580 \times 10^{-2}\) | \(1.0600 \times 10^{-15}\) | \(-1.0316 \times 10^{-2}\) | \(1.5580 \times 10^{-2}\) |
| f17      | 0.6721 | 0.3320 | 0.4236 | 0.0330 | \(3.979 \times 10^{-2}\) | \(8.4914 \times 10^{-2}\) | \(0.3979 \times 10^{-2}\) | \(8.4914 \times 10^{-2}\) |
| f18      | 23.2291 | 20.9353 | 1.8293e + 04 | 7.3490e + 04 | \(4.6200 \times 10^{-2}\) | \(11.3971 \times 10^{-2}\) | \(3.0000 \times 10^{-2}\) | \(1.2417 \times 10^{-2}\) |
| f19      | -3.63205 | 0.0829 | -3.3841 | 1.0401 | \(-3.7855 \times 10^{-2}\) | \(5.435 \times 10^{-2}\) | \(-3.8628 \times 10^{-2}\) | \(6.2486 \times 10^{-2}\) |
| f20      | -1.9270 | 0.4826 | -1.6300e-22 | 1.6299e-21 | \(-0.3251 \times 10^{-2}\) | \(9.803 \times 10^{-2}\) | \(-3.2709 \times 10^{-2}\) | \(0.0592 \times 10^{-2}\) |
| f21      | -8.2160 | 2.2302 | -1.01319 | 0.1073 | \(-6.0938 \times 10^{-2}\) | \(3.4405 \times 10^{-2}\) | \(-8.9806 \times 10^{-2}\) | \(2.1562 \times 10^{-2}\) |
| f22      | -8.2084 | 2.5139 | -1.04007 | 0.0088 | \(-6.7265 \times 10^{-2}\) | \(3.6376 \times 10^{-2}\) | \(-8.7020 \times 10^{-2}\) | \(2.4919 \times 10^{-2}\) |
| f23      | -5.75 | 0.00867 | -1.05354 | 0.0072 | \(-6.2799 \times 10^{-2}\) | \(3.7142 \times 10^{-2}\) | \(-7.324 \times 10^{-2}\) | \(2.7176 \times 10^{-2}\) |
| f24      | -1.8439e + 02 | 13.8362 | -1.8673e + 02 | 4.1968e-04 | \(-1.8686e + 02\) | \(0.1955 \times 10^{-2}\) | \(-1.8673 \times 10^{-2}\) | \(2.1621 \times 10^{-2}\) |

PIO: pigeon-inspired optimization; QUATRE: QUasi-Affine TRansformation Evolutionary; PSO: particle swarm optimization; QTPIO: QUasi-Affine TRansformation-Pigeon-Inspired Optimization.
Figure 1. Function model and comparison of four algorithms of PIO, PSO, QUATRE, and QT-PIO performance in test functions of single-modal.
and the curves of multi-modal functions is shown in Figure 2. Figure 3 holds up the curves of PIO, QUATRE, PSO, QT-PIO with the best final fitness value obtained minimum for the selected fixed-dimension multimodal functions in experiments. In addition, Figure 1, Figure 2 and Figure 3 display the function search space of particles is used for optimization.

As known from Figure 1, the curve of the QT-PIO algorithm not only converges fastest but also has the best optimal solution. This means that it performs best under all selected seven unimodal functions. As a consequence, we could say the QT-PIO algorithm optimizes the original function in the case of unimodal functions. We select six multimodal functions for testing, Figure 2 indicates that the QT-PIO algorithm achieves the best optimal solution as soon as possible under the five functions of six multi-modal functions. Figure 3 shows that QT-PIO achieves a fast convergence speed.

In accordance with the comparison of Figures 1, 2, 3, and Table 6, there is no doubt the convergence speed of QT-PIO is distinctly faster and can achieve better optimal value. Strictly, under the selected test functions modal the proposed algorithm can be said to be beautiful.

On the whole, QT-PIO is excellent and effective compared with before. Local optimization in the original PIO algorithm is solved well in the proposed algorithm, and the diversity of QT-PIO algorithm is improved effectively.

**Experiment for DV-Hop in WSNs problem**

The research of positioning technology has far-reaching significance. Compared with traditional sensors that have limitations, WSNs have numerous advantages, such as smaller size, less energy consumption, simple organization, no special personnel required, and high fault tolerance. Not only can it reduce network deployment time, cut down deployment costs, but it can also be used in a number of areas where traditional sensors cannot.

WSN is a self-organized, multi-hop distributed network composed of randomly deploying sensor nodes that communicate wirelessly in the monitored area. According to whether the distance between practical nodes requires to be measured during the locating process, the localization algorithm can be classified into a range-free location algorithm and range-based positioning algorithm. The DV-Hop is a non-ranging location algorithm based on distance vector routing in order to avoid direct measurement of the distance.
between nodes. Distance vector routing is based on the distance of the destination to determine the best path.

**Incipient DV-Hop**

Pristine positioning algorithm obtains the minimum hop-count by distance vector routing between the nodes whose location is unknown and the beacon node first. Next, dividing the distance between each beacon node by the total number of hops gets the average distance per hop. The average distance $h_{\text{aved}}$ of per hop is calculated in equation (20)

$$h_{\text{aved}} = \frac{\sum_{i=1}^{n} \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}}{\sum_{i=1}^{n} \text{hop}_{ij}},$$

(20)

where $\text{hop}_{ij}$ is the hop-count from the anchor node $i$ to the anchor node $j$.

And then the average distance of per hop is multiplied by the minimum hop-count between this unknown node and the beacon node, hence the obtained product is taken as the estimated distance between this unknown node and the beacon node. In the end, the location estimation of the unknown node is obtained by means of the maximum likelihood estimation method. In this way, the error of state estimation would be tested by the objective function as equation (21)

$$f(x, y) = \min \left( \sum_{i=1}^{n} \frac{1}{\text{hop}_{ai}} \left( \sqrt{(x - w_i)^2 + (y - z_i)^2 - d_i} \right)^2 \right).$$

(21)

There $n$ is the amount of all anchor nodes, $d_i$ signifies estimated distance between the unknown node and anchor node, $(x, y)$ denotes the estimated position coordinate of unknown nodes in the intelligence algorithm combined with DV-Hop, $(w_i, z_i)$ means the position coordinate of the anchor node.

**Our algorithm for optimizing DV-Hop**

Although DV-Hop has the characteristics of a simple method, the requirements for accuracy are getting higher with the needs of life and production. Therefore, people often combine intelligent computation with it to locate. In this article, we make use of QT-PIO to optimize DV-Hop, which improves the precision of positioning and reduces the error. In the optimized DV-Hop, first calculates the minimum value of hops among nodes and figures up the average hop

---

**Figure 3.** Function model and comparison of four algorithms of PIO, PSO, QUATRE, and QT-PIO performance in randomly fixed-dimension multimodal functions.
distance of each beacon node and then the hop-count is used to estimate the distance. At last, our novel QT-PIO algorithm is applied to estimate the position coordinates of unknown nodes. The main problem of positioning in the WSNs is to reduce the positioning error, here it uses the novel algorithm to optimize. The experiment considers the two-dimensional best final position obtained in the QT-PIO algorithm as the approximated location of the unknown nodes.

The detailed procedure is as below:

1. Set parameters of QT-PIO and initialize distance and hop-count matrix among nodes;
2. Generate the minimum hop-count among all the nodes;
3. Calculate average hop distance of beacon nodes using equation (21);
4. $d_i$ is figured out;
5. Employ QT-PIO to gain the two-dimensional variable.

**Optimal experiment**

The parameters needed in our algorithm for optimizing DV-Hop are flowing: the edge length of the area $A = 100$, the total amount of all nodes $N_{Amount} = 350$, all anchor nodes are 70, wireless communication distance $S = 30$, dimension $D = 2$, individual $ps = 30$, maximum iteration is set to 600. QT-PIO algorithm runs 20. Here we compare the performance qualities of the advanced DV-Hop optimized by QT-PIO algorithm with the primary DV-Hop, DV-Hop optimized by PIO algorithm, DV-Hop optimized by PSO algorithm, and DV-Hop optimized by QUATRE algorithm.

In the course of combining evolutionary algorithms with sensor localization, the two-dimensional optimal solution $(x_i, y_i)$ signifies the $i^{th}$ unknown node. The errors of the position of estimated unknown nodes are calculated by equation (22). Table 7 shows the comparison of positioning errors and accuracy that come out of equation (23)

$$
\left\{ \begin{array}{l}
er_i = \sqrt{(x_i - x_i^1)^2 + (y_i - y_i^1)^2}, \\
\quad e_r = \frac{\sum(e_{ri})}{m},
\end{array} \right. \quad (22)
$$

where $m$ is the total quantity of unknown nodes and $(x_i^1, y_i^1)$ represents the practical coordinate of the $i^{th}$ unknown node

$$
\left\{ \begin{array}{l}
er_i = \sqrt{(x_i - x_i^1)^2 + (y_i - y_i^1)^2}, \\
\quad accuracy = \frac{\sum(e_{ri})}{mvS},
\end{array} \right. \quad (23)
$$

**Table 7.** Comparison of error and accuracy results.

| Measurement index | QT-PIO | DV-Hop | PIO | PSO | QUATRE |
|-------------------|--------|--------|-----|-----|--------|
| Error             | 4.2395 | 9.7851 | 5.0124 | 4.7376 | 4.2542 |
| Accuracy          | 0.1415 | 0.3262 | 0.1589 | 0.1494 | 0.1418 |
| Aveaccuracy       | 0.1416 | 0.1631 | 0.1597 | 0.1418 | 0.1418 |

QT PIO: QUasi-Affine TRansformation-Pigeon-Inspired Optimization; PSO: particle swarm optimization; QUATRE: QUasi-Affine TRansformation Evolutionary.

**Figure 4.** Comparison of PIO, PSO, QUATRE, and QT-PIO combining with DV-Hop performance in specific test functions.

Sun et al. 13
where $S$ is the communication distance of nodes.

Figure 4 depicts the comparison of errors of the applied QT-PIO, PSO, PIO, QUATRE after running 20. When scanning the errors and accuracy of four algorithms applied to DV-Hop, the QT-PIO algorithm could reduce the errors of estimation effectively and have a great effect on the positioning precision.

Distinctly observed from above that the results of the QT-PIO under the given testing function for the DV-Hop positioning algorithm in WSNs is preferable than the other three algorithms in the process of comparison.

Conclusion

To improve the capability and performance of PIO, this study proposed a new optimization algorithm, which mainly applies the co-evolutionary $M$ and mutation $B$ in QUATRE algorithm to PIO algorithm, and enables individuals to update their positions in the form of matrix in the process of optimization. This learning strategy not only merely raises the diversity of PIO but also effectively avoids the problem that pigeons of the PIO algorithm plunge into a local optimal solution. The curve of registration results demonstrates the converging speed of this update strategy model. Furthermore, the rate of convergence is faster than the traditional model, and this model has good practicability. Later we use the new optimized algorithm for DV-Hop positioning, which further improves the error caused by positioning and improves the accuracy. Our algorithm supplies an effective optimization mechanism for going into WSNs and the application of WSNs. The proposed QT-PIO algorithm may be further improved by adopting some efficient methods.29-30

Acknowledgements

This class file was developed by Sunrise Setting Ltd, Brixham, Devon, UK.

Website: http://www.sunrise-setting.co.uk

Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This work is supported by the National Natural Science Foundation of China (61872085), the Natural Science Foundation of Fujian Province (2018J01638) and the Fujian Provincial Department of Science and Technology (2018Y3001).

ORCID iDs

Xiao-Xue Sun https://orcid.org/0000-0002-3986-6368
Jeng-Shyang Pan https://orcid.org/0000-0002-3128-9025
Shu-Chuan Chu https://orcid.org/0000-0003-2117-0618
Pei Hu https://orcid.org/0000-0003-3711-4952
Ai-Qing Tian https://orcid.org/0000-0003-0808-2015

References

1. Duan H and Qiao P. Pigeon-inspired optimization: a new swarm intelligence optimizer for air robot path planning. Int J Intell Comput Cybernet 2014; 7(1): 24–37.
2. Li H, Li H, Chen X, et al. An improved pigeon-inspired optimization for clustering analysis problems. Int J Comput Intell Appl 2017; 16(2): 809–818.
3. Wang H, Rahnamayan S, Sun H, et al. Gaussian bare-bones differential evolution. IEEE Tran Cybernet 2013; 43(2): 634–647.
4. Wang H, Jin Y, Sun C, et al. Offline data-driven evolutionary optimization using selective surrogate ensembles. IEEE Trans Evolut Comput 2018; 23(2): 203–216.
5. Kong L, Pan J-S, Tsai P-W, et al. A balanced power consumption algorithm based on enhanced parallel cat swarm optimization for wireless sensor network. Int J Distrib Sens Net 2015; 11(3): 1–10.
6. Kumar S and Lobiyal D. An advanced DV-Hop localization algorithm for wireless sensor networks. Wirel Pers Commun 2013; 71(2): 1365–1385.
7. Chen X and Zhang B. Improved DV-Hop node localization algorithm in wireless sensor networks. Int J Distrib Sens Net. Epub ahead of print 1 August 2012. DOI: 10.1155/2012/213980.
8. Nguyen T-T, Pan J-S and Dao T-K. An improved flower pollination algorithm for optimizing layouts of nodes in wireless sensor network. IEEE Access 2019; 7: 75985–75998.
9. Meng Z, Pan J-S and Tseng K-K. PaDE: an enhanced algorithm with novel control parameter adaptation schemes for numerical optimization. Knowl-based Syst 2019; 168: 80–99.
10. Kung H-Y, Chen C-H and Ku H-H. Designing intelligent disaster prediction models and systems for debris-flow disasters in Taiwan. Expert Syst Appl 2012; 39(5): 5838–5856.
11. Duan H and Qiu H. Advancements in pigeon-inspired optimization and its variants. Sci China Inform Sci 2019; 62: 70201.
12. Wang J, Gao Y, Wang K, et al. An affinity propagation-based self-adaptive clustering method for wireless sensor networks. Sensors 2019; 19(11): 2579.
13. Liu N, Pan J-S and Nguyen T-T. A bi-population QUasi-Affine TRansformation evolution algorithm for global optimization and its application to dynamic deployment in wireless sensor networks. EURASIP J Wirel Comm. Epub ahead of print 1 December 2019. DOI: 10.1186/s13638-019-1481-6.
14. Chu S-C, Xue X, Pan J-S, et al. Optimizing ontology alignment in vector space. J Intern Technol 2020; 21: 15–22.
15. Meng Z, Pan J-S and Xu H. QUasi-Affine TRansformation evolutionary (QUATRE) algorithm: a cooperative swarm based algorithm for global optimization. *Knowl-based Syst* 2016; 109: 104–121.
16. Yin X, Zhang K, Li B, et al. A task allocation strategy for complex applications in heterogeneous cluster–based wireless sensor networks. *Int J Distrib Sens Netw*. Epub ahead of print 21 August 2018. DOI: 10.1177/1550147718795355
17. Pan J-S, Hu P and Chu S-C. Novel parallel heterogeneous meta-heuristic and its communication strategies for the prediction of wind power. *Proces* 2019; 7(11): 845.
18. Kennedy J and Eberhart R. Particle swarm optimization (PSO). *IEEE Int Conf Neu Netw* 1995; 4: 1942–1948.
19. Wang J, Ju C, Gao Y, et al. A PSO based energy efficient coverage control algorithm for wireless sensor networks. *Comput Mater Contin* 2018; 56(3): 433–446.
20. Chu S-C, Tsai P-W and Pan J-S. Cat Swarm Optimization. *PRICAI* 2006; 4099: 854–858.
21. Sun C, Jin Y, Cheng R, et al. Surrogate-assisted cooperative swarm optimization of high-dimensional expensive problems. *IEEE Tran Evolut Comput* 2017; 21(4): 644–660.
22. Chang J-F, Chu S-C, Roddick JF, et al. A parallel particle swarm optimization algorithm with communication strategies. *J Inform Sci Eng* 2005; 21(4): 809–818.
23. Pan J-S, Kong L, Sung T-W, et al. A clustering scheme for wireless sensor networks based on genetic algorithm and dominating set. *J Inter Technol* 2018; 19(4): 1111–1118.
24. Chen C-H, Lee C-A and Lo C-C. Vehicle localization and velocity estimation based on mobile phone sensing. *IEEE Access* 2016; 4: 803–817.
25. Wang J, Gao Y, Yin X, et al. An enhanced PEGASIS algorithm with mobile sink support for wireless sensor networks. *Wirel Comm Mob Comput* 2018; 2018: 1–9.
26. Wang J, Gao Y, Liu W, et al. An improved routing schema with special clustering using PSO algorithm for heterogeneous wireless sensor network. *Sensors* 2019; 19(3): 671.
27. Pan J-S, Kong L, Sung T-W, et al. α-fraction first strategy for hierarchical model in wireless sensor networks. *J Inter Technol* 2018; 19(6): 1717–1726.
28. Wang J, Gu X, Liu W, et al. An empower hamilton loop based data collection algorithm with mobile agent for WSNs. *Hum-cent Comput Info Sci* 2019; 9(1): 18.
29. Pan J-S, Lee C-Y, Sghaier A, et al. Novel systolization of subquadratic space complexity multipliers based on toeplitz matrix-vector product approach. *IEEE Tran VLSI Syst* 2019; 27: 1614–1622.
30. Xue X, Chen J and Yao X. Efficient user involvement in semi-automatic ontology matching. *IEEE TETCI*. Epub ahead of print 13 December 2018. DOI: 10.1109/TCI.2018.2883109.