Application of A Multi-Species Optimizer in Ubiquitous Computing for RFID Networks Scheduling

Hanning Chen1, 2, Yunlong Zhu1, Kunyuan Hu1, 2, Ben Niu1, 2
1 Shenyang Institute of Automation, Chinese Academy of Sciences, 110016, Shenyang, China,
2 School of Graduate, Chinese Academy of Sciences, 100039, Beijing, China
{chenhanning, ylzhu}@sia.cn

Abstract

Devising switching schemes for networks of colliding and correlated RFID readers is a core challenge in the reliable operation of RFID augmented ubiquitous environment. In this paper, a novel optimal scheduling scheme for RFID networks using a symbiotic multi-species particle swarm optimizer (SMSO) is presented. The SMSO, which inspired by the biological phenomenon of symbiosis in nature, can remarkably enhance the convergence and accuracy of the standard particle swarm optimizer (SPSO) when applied to perform the nonlinear optimization problems. Numerical simulations of three benchmark functions are used to test the performance of SMSO. Furthermore, simulation on RFID reader networks architecture is given to illustrate the effectiveness of the proposed scheduling scheme.

1. Introduction

Mark Weiser in his famous article describes a vision of ubiquitous computing in which a number of computers and sensors are deeply integrated into our daily lives and users can access information at any time and in any place [1]. RFID is being developed as an essential elemental technology for realizing this vision, and has been demonstrated by many researchers over the years, such as the prototypes of magic medicine cabinet and magic wardrobe [2], the augmentation of the operation system in container terminals [3] and the smart shelves [4, 5]. However, there also some weaknesses associated with the deployment of the RFID-based application. Particularly in a multi-tag and multi-reader environment, the phenomenon of false negative reads occurs, where a tag that is present is not detected.

PSO is a newly developed evolutionary technique [11]. Due to its simplicity in implementation and high computational efficiency in solving optimization problems, it has already come to be widely used in many areas. However, it was pointed out that PSO suffers from premature convergence, tending to get stuck in local optima.

In this paper, Inspired by symbiosis [7], a novel multi-species optimization algorithm (SMSO) is proposed to address the shortcomings of SPSO. The SMSO extends the dynamics of the standard PSO algorithm by adding a significant ingredient, which takes into account symbiotic coevolution between species. The SMSO is utilized as the optimization technique to solve the problem of failed RFID tags read. In particular, we address the problems of potential collisions between readers, and the correlations between the various RFID readers that cause the faulty or missing reads.

The rest of this paper is organized as follows. Section 2 describes the problem of RFID reader networks scheduling. In Section 3, three variant versions of SMSO are proposed and tested on a set of mathematic benchmark functions. Section 4 provides the detailed design algorithm of optimal scheduling for RFID reader networks using SMSO. Simulation results are presented in section 5. Finally, conclusions are drawn in Section 6.

2. RFID readers scheduling

We extract techniques started with the application of graph theoretic tools to the problem of assigning radio frequency spectrum to a set of radio frequency transmitters, such as cellular telephone base stations [6]. The other method of collision avoidance is time scheduling. In this approach, readers with overlapping fields are fired at different times so that they do not collide. Both of these allocations (spectral and temporal) can be modeled as graph partitioning problems (GPP). Thus graphical models found their way into the analysis of RFID reader collision problems.

Given a collection of RFID readers laid out in some manner, we can construct the associated collision graph G=(V,E) where each vertex v ∈ V corresponds to a RFID reader and each edge e ∈ E indicates that those two readers can be operated in parallel (i.e., there are no
collisions between these two readers). For example, the collision graph corresponding to the RFID readers layout of Fig 1(a) is given in Fig 1(b). GPP is to partition a graph \( G \) into \( k \) subgraphs such that the number of edges connecting nodes in different subgraphs is minimized, and the number of edges connecting nodes of the same subgraph is maximized.

The frequency allocation problem for networks of RFID readers is to allocate frequencies to various readers. I.e., when two readers lie in each other's interference region, they are given different frequencies. Clearly this problem reduces to graph partitioning problems (GPP) [10].

The time scheduling problem in the RFID system is to find the optimal sequence for firing the various RFID readers. In our model RFID applications, there are two constraints which limit the parallel operation of readers. The first is reader interference or collision. At this point, this problem looks like the frequency allocation problem, except that the allocation is done along the time axis. Interfering readers are allotted non-overlapping periods of time so as to avoid collision between them. Thus it may appear that this problem also reduces to the graph coloring problem. The second constraint is the overall transaction time. This makes clear that partitioning the graph cannot completely solve the time scheduling problem, since it also requires minimizing the total transaction time of the RFID networks, whereas the frequency allocation problem is not.

**3. Proposed algorithm**

**3.1 Review of the standard PSO**

In early versions of PSO formulae, each particle is represented as \( \bar{x}_i = (x_{i1}, x_{i2}, \ldots, x_{in}) \) and records its previous best position represented as \( P_i = (P_{i1}, P_{i2}, \ldots, P_{in}) \), which is also called \( p_{best} \).

The index of the best particle among all the particles in the population is represented by the symbol \( g \), and \( p_{g} \) is called \( g_{best} \). At each iteration, The velocity of particle \( i \), represented as \( \bar{v}_i = (v_{i1}, v_{i2}, \ldots, v_{in}) \), are manipulated according to the following equations:

\[
\begin{align*}
\bar{v}_i &= \chi (w \bar{v}_i + c_1 r_1 (P_{id} - x_{id}) + c_2 r_2 (p_{gd} - x_{id})), \\
x_{id} &= x_{id} + \bar{v}_{id}.
\end{align*}
\]

Where \( \chi \) is known as constriction coefficient, \( w \) is inertia weight; \( c_1 \) and \( c_2 \) are learning rates; and \( r_1, r_2 \) are two random vectors uniformly distributed in \([0, 1]\).

**3.2. Symbiotic multi-species optimizer**

We now describe the SMSO algorithm for evolving symbiotic coadapted species. The whole population is divided into \( n \) species to model symbiosis in the context of the evolving ecosystems (for convenience, each species has the same population size \( m \)). The blackboard model [8] is introduced into SMS-PSO for realizing symbiotic mutualism and cooperative coevolution of dissimilar species. Our symbiotic coevolution model is shown in Figure 2. Each species in SMSO represents an agent in blackboard model and posts its species best position to and reads best positions attained by other species so far from the blackboard. This mechanism performs in each generation and acts as a “shared memory” for the whole population. Then each species evolves based on its own knowledge and also the knowledge of other species that learned from the blackboard.

To realize this mechanism, this paper proposes a significant modification to the standard PSO velocity update equation. In each generation, particle \( i \) in species \( n \) will evolve according to the following equations:

\[
\begin{align*}
\bar{v}_{id} &= \chi (w \bar{v}_{id} + c_1 r_1 (P_{id} - x_{id}) + c_2 r_2 (p_{gd} - x_{id}) + c_3 r_3 (P_{gd} - x_{id})), \\
x_{id} &= x_{id} + \bar{v}_{id},
\end{align*}
\]

The new term adding into the velocity update equation, \( c_3 r_3 (P_{gd} - x_{id}) \), represents the symbiotic coevolution.
among dissimilar species and takes account the “shared memory” in the whole population. Where \( n \) is the species number, \( c_3 \) is the “symbiotic learning rate”, \( r_3 \) is a uniform random sequence in the range \([0, 1]\), and \( p_{sd} \) is the current best one select from the positions each species posted on the blackboard. In this way, each particle obtains two dynamic characteristics which govern its movement: one is inspired by the model of sociological behavior within one species, and the other is inspired by the model of symbiotic coevolution among dissimilar species. The pseudocode for the SMSO is listed in Table 1.

### Table 1 Pseudocode of SMSO algorithm

**INITIALIZE.** Randomize positions and velocities of \( n \times m \) particles in search space. Divide whole population into \( n \) species with \( m \) particles randomly;

**WHILE** (the termination conditions are not met)

FOR (each species \( n \)) IN PARALLEL

Choose the current best one as \( p_{sd} \) from the positions all species posted on the blackboard;

FOR (each particle \( m \) of species \( n \))

Update the velocity and position using equations (3) and (4)

END FOR

Posts its species best position to the blackboard.

END FOR IN PARALLEL

END WHILE

### 3.2. Simulation results of benchmark function

All the algorithms are tested on three 30 dimensional benchmark functions to show solution quality and convergence rate. Dimensions, initialization ranges and global optimum of each function are listed in Table 2. The functions are listed below:

1. **Sphere function**

\[
f_1(x) = \sum_{i=1}^{n} x_i^2
\]

2. **Rosenbrock function**

\[
f_2(x) = \sum_{i=1}^{n} 100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2
\]

3. **Griewank function**

\[
f_3(x) = \sum_{i=1}^{n} (x_i^2 - 10\cos(2\pi x_i)) + 10
\]

### Table 2. Parameters of the test functions

<table>
<thead>
<tr>
<th>Func.</th>
<th>Dim.</th>
<th>Search area</th>
<th>Minimum</th>
</tr>
</thead>
<tbody>
<tr>
<td>( f_1 )</td>
<td>30</td>
<td>([-100,100]^D)</td>
<td>0</td>
</tr>
<tr>
<td>( f_2 )</td>
<td>30</td>
<td>([-2.048,2.048]^D]</td>
<td>0</td>
</tr>
<tr>
<td>( f_3 )</td>
<td>30</td>
<td>([-600,600]^D]</td>
<td>0</td>
</tr>
</tbody>
</table>

### Table 3. Parameter setting

<table>
<thead>
<tr>
<th>Type</th>
<th>SMSO-w1</th>
<th>SMSO-w2</th>
<th>SMSO-cf</th>
</tr>
</thead>
<tbody>
<tr>
<td>( n )</td>
<td>12</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>( m )</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>( \chi )</td>
<td>1</td>
<td>1</td>
<td>0.729</td>
</tr>
<tr>
<td>( W )</td>
<td>0.9 to 0.4</td>
<td>0.9 to 0.4</td>
<td>—</td>
</tr>
<tr>
<td>( c_1 )</td>
<td>1.0</td>
<td>2.0</td>
<td>1.3667</td>
</tr>
<tr>
<td>( c_2 )</td>
<td>1.0</td>
<td>2.0</td>
<td>1.3667</td>
</tr>
<tr>
<td>( c_3 )</td>
<td>2.0</td>
<td>1.0</td>
<td>1.3667</td>
</tr>
</tbody>
</table>

### Table 4. Parameter setting

<table>
<thead>
<tr>
<th></th>
<th>SMSO-w1</th>
<th>SMSO-w2</th>
<th>SMSO-cf</th>
<th>SMSO-w1</th>
</tr>
</thead>
<tbody>
<tr>
<td>( f_1 )</td>
<td>2.3054e-060</td>
<td>9.4597e-004</td>
<td>3.3982e-129</td>
<td>2.1761e-021</td>
</tr>
<tr>
<td>( f_2 )</td>
<td>1.5989e+001</td>
<td>2.8309e+001</td>
<td>5.7666e-001</td>
<td>2.5278e+001</td>
</tr>
<tr>
<td>( f_3 )</td>
<td>1.1533e-002</td>
<td>2.4985e-001</td>
<td>1.9125e-002</td>
<td>1.6335e-002</td>
</tr>
</tbody>
</table>

For standard PSO, population size is set at 60. The parameters were set to the values \( c1 = c2 = 2 \), and \( \chi = 1 \). A linearly inertia weight \( W \) starting at 0.9 and ending at 0.4 was used. The maximum velocity of each particle was set to be 20% of the search space. For SMSO, except common parameters used in PSO, there are additional parameters setting strategies in SMSO need to be specified. The parameters setting for three SMSO variations are summarized in Table 3.

The experiment run 50 times respectively for each algorithm on each benchmark function and max generation is set at 2000. The mean function values obtained by all algorithms are present in Table 4. Figure 3-5 present the evolution process for all algorithms.
according to the reported results in Table 4. From the results we can conclude that the SMSO algorithm is markedly outperformed standard PSO for all the test functions in terms of accuracy and convergence speed.

![Fig. 3. Sphere function](image)

![Fig. 4. Rosenbrock function](image)

![Fig. 5. Griewank function](image)

4. Optimal scheduling for RFID networks

The overall procedure of using SMSO to solve the problem of scheduling RFID reader networks is described as follows:

(1) Particle representation

In our work the direct encoding scheme is applied to encode the individuals. The possible solutions are represented as dynamic particle dimension according to different time steps where:

Particle dimension = the number of readers in certain time step.

Each element in the dimension is corresponding to the absence of the particular reader, whose entries can only be “0” or “1”. A bit “0” in an individual indicated the absence of the corresponding reader. Otherwise a bit “1” in an individual indicated the presence of the corresponding reader.

An example of a particle representing a collision graph of 10 RFID readers along the time axis is show in Fig. 6.

![Fig. 6. The particle presenting of RFID readers](image)

(2) Initialization

The m*n individuals forming the population should be randomly generated and each represents a possible solutions.

(3) Fitness function design

To evaluate the performance of an individual, a predefined fitness function should be formulated [9]. The fitness function takes into account four parameters. The fitness is calculated as the reciprocal of \( C \) as follows:

\[
C = w_1 \times (N) + w_2 \times (T) + w_3 \times (W) + w_4 \times (C) - p \times (N)
\]  

(8)

\[
f = \frac{1}{C}
\]  

(9)

\( N \) is the number of readers, \( T \) is the transaction time of the partition and can be calculated as:

\[
T = \max (\sum t_i)
\]  

(10)

Where \( t_i \) is the transaction time of the \( i \)th reader that forming the partition.

\( W \) is the weight attached to this group of readers and can be calculated as:

\[
W = \frac{1}{N} \sum t_i
\]  

(11)

Parameter \( W \) is to minimize the variation in reader transaction time within the partition. Besides minimizing the range of transaction values in the partition, it also saves readers with lower transaction times for further partitioning and reduces overall transaction time of the system.

\( C \) is the summation of all the possible collision that the members of the partition have with the readers still remaining in the graph to be partitioned in next time step.
$w_1, w_2, w_3, w_4$ are the weights given to each term of the fitness function and $w_2 \geq w_1 \geq w_4 \geq w_3$ implying the importance of each term respectively.

4. Update collision and transaction time
After the velocity and position update, the individuals associated with both the collision and transaction times are updated to produce new best-performing individuals.

5. Termination condition
The proposed algorithm is performed until the fitness is small enough, or a pre-determined number of iterations are passed. It is expected also, after a certain number of iterations, all the readers will be grouped and the optimal scheduling can be obtained.

The particle representation is binary and hence binary SMSO algorithm has been used to evolve the subgraph. The binary algorithm is very similar to the continuous model in terms of velocity update as in equation (3). The difference is in position the update equation, which is given by:

$$\begin{align*}
\text{if } (\text{rand}() < S(v_{id})) & \text{ then } x_{id} = 1; \text{ else } x_{id} = 0 \\
S(v_{id}) &= \frac{1}{1+\exp(-v_{id})}
\end{align*}$$

The pseudocode for implementing our algorithm is listed in Table 5.

**Table 5** Pseudocode of the proposed algorithm

```
INITIALIZE. Randomize positions and velocities of \( n \times m \) particles in search space. Divide whole population into \( n \) species with \( m \) particles randomly;
WHILE (readers still left for scheduling)
  WHILE (the termination conditions are not met)
    FOR (each species \( n \)) IN PARALLEL
      Choose the current best one as \( \nu^*_{sd} \) from the positions all species posted on the blackboard;
      FOR (each particle \( m \) of species \( n \))
        Update the velocity and position using equations (3), (12) and (13).
      END FOR IN PARALLEL
    END FOR IN PARALLEL
  END WHILE
END WHILE
Delete subgraph generated by SMSO from collision graph
Update collision graph
Update transaction times
```

5. Simulation test and analysis
In this section, a RFID reader network application, including 10 readers is scheduled to validate the capability of the proposed method.

Table 6 list the processing times for this 10 readers networks and the collision graph are shown as in Fig 6. The simulation results that obtained by our algorithm are listed in Table 7.

In simulation test, the parameters of the problem are set as: \( w_1 = 20, w_2 = 1000, w_3 = 1 \times 10^{-4}, \) and \( w_4 = 0.1 \). The parameters of SMSO are same as section 3.2. The maximum iterations are set as 500.

It should be noted that the SMSO algorithm can constantly find an optimal schedule results. In fact, with an increasing in the number of the readers (hence the degree of the graph), the problem of finding best solution becomes intractable. However, for the larger scale reader networks, our proposed method is able to find the optimal schedule results robustly and consistently.

6. Conclusions and Future Work
This paper is devoted to giving a new strategy for scheduling reader networks in RFID-based ubiquitous computing environment. A symbiotic mechanism based algorithm, symbiotic multi-species optimizer, is proposed to search through space for an optimization problem.

Simulation results on both mathematical benchmark functions and a real–world problem (i.e., the optimal scheduling for RFID reader networks) show that the SMSO algorithm offers more robust and consistent performance in term of both solution quality and convergence rate. In fact our proposed method is suitable for scheduling large scale RFID reader networks.
Table 6 Processing times of 10 readers

<table>
<thead>
<tr>
<th>Reader</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reader</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
</tr>
</tbody>
</table>

Table 7 Schedule result of 10 readers

<table>
<thead>
<tr>
<th>Time step</th>
<th>Readers</th>
<th>Processing time</th>
<th>Total time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5 6 7 8</td>
<td>15.9297</td>
<td>15.9297</td>
</tr>
<tr>
<td>2</td>
<td>3 4</td>
<td>14.7618</td>
<td>30.6915</td>
</tr>
<tr>
<td>3</td>
<td>110</td>
<td>14.7606</td>
<td>45.4521</td>
</tr>
<tr>
<td>4</td>
<td>9</td>
<td>9.2377</td>
<td>54.6898</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>3.9499</td>
<td>58.6397</td>
</tr>
</tbody>
</table>

Acknowledgements

This work is supported by the National Natural Science Foundation of China (Grant No. 70431003) and the national 863 project under contract No20060104A1118.

7. References