An improved deployment algorithm for wireless sensor networks based on Particle Swarm Optimization

DING Shuxin

Beijing Institute of Technology
2013-07-01
Outline

• INTRODUCTION
• PROBLEM FORMATION
• PROPOSED ALGORITHM
• SIMULATION RESULTS
• CONCLUSION & FUTURE WORK
INTRODUCTION

• Wireless Sensor Networks (WSNs)
  • Some sensors: small in size, low in power and cost
  • Short distance communication
  • Communicating without wires

• Some applications of WSNs
  • District/Environment monitoring
    • Measure the environment conditions to forecast disasters and to give early warnings before the occurrences
    • Deployed in places which are hard to reach (volcanic area monitoring)
Challenges of WSNs

• Sensors deployment optimization is an important issue
  • Limited communicating range and lifetime

• Make full use of limited detecting resources
  • Sensors should be placed within certain range
Sensors deployment

• Given conditions
  • Limited number of sensors with limited detecting range
  • Given region of interest (ROI)

• Possible objectives of deployment
  • Maximum coverage
  • Minimum overlap coverage

• In this paper, just consider maximum coverage
Coverage Problem

• Coverage rate evaluate the performance of the WSNs deployment
  • The positions of the sensors determines the coverage rate
• Detecting probability

probability function of $i^{th}$ sensor $s_i(x_i, y_i)$ for point $P(x, y)$

$$c_{x,y}(s_i) = \begin{cases} 0 & \text{if } r_i < d \\ 1 & \text{if } r_i \geq d \end{cases}$$

$$d = \sqrt{(x-x_i)^2 + (y-y_i)^2}$$

probability function of $n$ sensors for point $P(x, y)$

$$c_{x,y} \left( \sum_{i=1}^{n} s_i \right) = 1 - \prod_{i=1}^{n} \left( 1 - c_{x,y}(s_i) \right)$$
• Grids are used to determine the coverage rate

Fig. 1 3 sensor nodes deployed in gridded ROI: (a) grid size: 0.5 × 0.5; (b) grid size: 0.25 × 0.25

The grid size should be chosen carefully which balances the calculation time and accuracy.

• Coverage rate

\[
R = \frac{\sum_{j=1}^{n_p} c_{x_j, y_j} \left( \sum_{i=1}^{n} S_i \right)}{n_p \sum_{i=1}^{n} S_i} \quad \text{detecting probability of } \sum_{i=1}^{n} S_i \text{ grid points}
\]

\[
R = \frac{n_p}{\sum_{i=1}^{n} S_i} \quad \text{the number of grid points in the ROI}
\]

Objective: maximize the coverage rate
Particle Swarm Optimization

- Particle Swarm Optimization (PSO) is an algorithm based on the social behavior of a flock of birds.
- The particles are moving in the searching space according to their former speed, their experience and the experience of their neighbors around.
- Each particle represents a potential solution of the optimization.
- More particles, more accurate.
- A particle for n sensors can be represented as \((x_1, y_1, x_2, y_2, x_3, y_3, \ldots, x_n, y_n)\)
  - 2n dimension: every 2 components represents a sensor’s position
Particle Swarm Optimization

• Evolution equation

\[ v_{id}(t + 1) = w \times v_{id}(t) + c_1 \times \text{rand()} \times (p_{ibest} - x_{id}) + 
  c_2 \times \text{rand()} \times (p_{gbest} - x_{id}) 
\]

\[ x_{id}(t + 1) = x_{id}(t) + v_{id}(t + 1) \]

- \( w \): inertia factor (0.9-0.4) \( v_{id}(t) \): former velocity
- \( p_{ibest} \): individual best position \( p_{gbest} \): global best position

• Fitness function (coverage rate) evaluates the position quality of a particle, and \( p_{ibest} \) and \( p_{gbest} \) are replaced according to it.

• PSO stops after maximum iterations.

• Local optimal for a high-dimensional optimization
  • Some components move closer to the optimal position while others may move away from the optimal, while it gives a better solution for larger coverage rate. “two step forward, one step back”.
PROPOSED ALGORITHM

• D-PSO algorithm based on PSO

• Evolution equation

\[ v_{id}(t + 1) = c_0 \times \text{randn()} + c_1 \times \text{rand()} \times (p_{ibest} - x_{id}) + c_2 \times \text{rand()} \times (p_{gbest} - x_{id}) \]

• \text{randn()} is a standard normal distributed function

• \( c_0 \) is set according to the numbers of the sensors, the sensing range and the space size

• No former velocity but a disturbance

• The update formula of the position is the same with PSO
• Difference of the sensor position motion tendency between PSO and d-PSO

![Diagram](image)

Fig. 2 Searching space between PSO algorithm (a) and d-PSO algorithm (b)

• PSO only considers more about the direction of its velocity

• The shadow area in d-PSO contains the shadow area in PSO

• \( c_0 \times randn() \) ranges from \([-3c_0, 3c_0]\) \[ c_0 = \frac{cL}{nr} \]

• \( c \) is a constant number(set as 1), ROI is a \( L \times L \) square area, \( n \) is the number of the sensors, \( r \) is the detection range of one sensor.

• This factor plays an important role for particles to converge to the global optimized solution. If this factor is too big or too small, this algorithm will perform badly.
• Difference of the sensor position motion tendency between PSO and d-PSO

![Diagram of searching space between PSO algorithm (a) and d-PSO algorithm (b)]

- PSO only considers more about the direction of its velocity
- The shadow area in d-PSO contains the shadow area in PSO
- $c_0 \times \text{randn()}$ ranges from $[-3c_0, 3c_0]$ \[ c_0 = \frac{cL}{nr} \]
  - $c$ is a constant number (set as 1), ROI is a $L \times L$ square area, $n$ is the number of the sensors, $r$ is the detection range of one sensor.
- when $x_{id} = p_{ibest} = p_{gbest}$, PSO stops ($v_{id} = 0$), while d-PSO with no velocity part, $v_{id} = c_0 \times \text{randn()}$, a $3c_0$ searching range as a circle
- Jump away from the optimal position for high dimensional optimization
SIMULATION RESULTS

- Deployment performance between PSO and d-PSO
  - 20 sensors; 20 particles; $c=0.75$; $c_1=c_2=1.4962$;
  - maximum iteration = 600;
  - $w=0.9$ - 0.4 (linearly decreasing with $t$) for PSO;
  - ROI: $40 \times 40m^2$; grid size: $1 \times 1$
(a) initial random placement 65.97%

(b) PSO algorithm 74.30%

(c) d-PSO algorithm 84.95%

d-PSO presents a better deployment solution than PSO
• PSO converges faster than d-PSO
• d-PSO seems to find better solution there are a large number of iterations
• disturbance always allows the particle to find better solutions

• 50 independent experiments
• d-PSO seems to be better in effectiveness and robustness than PSO

![Fig.4 The coverage rate of PSO and d-PSO during the iterations.](image)

Table 1. The average coverage rate, its standard deviation and execution time

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>PSO</th>
<th>d-PSO</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average coverage rate (%)</td>
<td>74.67</td>
<td>84.88</td>
</tr>
<tr>
<td>Standard Deviation (%)</td>
<td>2.04</td>
<td>1.08</td>
</tr>
<tr>
<td>Average execution time (s)</td>
<td>7.13</td>
<td>6.98</td>
</tr>
</tbody>
</table>
• Analysis of the particle numbers

• 50 independent experiments
• Particles: 1 to 40
• d-PSO algorithm has an acceptable solution when there are few particles
• PSO algorithm seems to be better when particle number is growing
• The execution time will cut down with fewer particles using d-PSO, with acceptable solutions.

Fig. 5 The coverage rate of PSO and d-PSO with different amount of particles
CONCLUSION & FUTURE WORK

• Conclusion
  • An improved deployment algorithm called d-PSO algorithm to solve the coverage problem of WSNs.
  • Better coverage rate; less particles to save time; better global searching ability than PSO.

• Future work
  • Study the convergence property of d-PSO and the factor of the disturbance.
  • More complicated description of ROI and sensors.
Thank you!