ADAPTIVE PARTICLE SWARM OPTIMIZATION FOR SOLVING
CONSTRUCTION SITE LAYOUT PROBLEM

Angelia M. Adrian¹*, Amalia Utamima¹, and Kung-Jeng Wang¹

¹Department of Industrial Management,
National Taiwan University of Science and Technology, Taipei City, 106, Taiwan, ROC

* D9901806@mail.ntust.edu.tw

Abstract

The positioning of temporary facilities on a construction site is important to enhance productivity and safety. PSO is a popular optimization algorithm that has been widely used in many fields to solve various problems. However it is also easily trapped in local optima. In this paper, an adaptive particle swarm optimization (APSO) is presented to guide algorithm to escape from local optima trap. An example problem from previous study was used to solve the problem in which m facilities are to be positioned to n available locations such that the total cost of construction and interactive cost due to facility layout constraints are minimized. The experiment results show that APSO is efficient in solving the construction site layout problem.

Keywords: construction site, site layout problem, particle swarm optimization.

1. INTRODUCTION

Site Layout is a crucial task in the construction industry and should be considered in the planning phase. A good facility layout will impact on money and timesaving, especially for large scale construction projects. The objective of construction site layout is to arrange the temporary facilities such as, job office, labor residence, warehouse, and batch plants, so that the construction work can be served satisfactorily with minimal costs and can improved safety and working environment.

A good site layout is important to promote safe and efficient operations, minimize travel time, decrease material handling, and avoid obstructing material and equipment
movements, particularly for the large case project (Tommelein et al., 1992). In recent years, researchers have try to address the facility layout problem by utilize the metaheuristic techniques. Yeh (1995) use artificial neural networks to improve a predetermined site layout. The model minimizes a total cost function that includes the cost of constructing a facility at the assigned location on site and the cost of interacting with other facilities. Li and Love (1998) used the genetic algorithms (GA) technique to solve site-level unequal-area facility layout problems. Mawdesley and Al-Jibouri (2003) presented a sequence based genetic formulation of the $m \times n$ facility layout problem. Zhang and Wang (2008) proposed a particle swarm optimization (PSO) based methodology to solve the construction site unequal-area facility layout problem. Because of the easy implementation of PSO, it has become a popular optimization algorithm that has been widely used in many fields to solve various problems. However it is also easily trapped in local optima. In this paper, an adaptive particle swarm optimization (APSO) is presented to guide algorithm to escape from local optima trapped.

2. THE CONSTRUCTION SITE LAYOUT PROBLEM

Site layout problem has been term as Quadratic Assignment Problem (QAP) which is categorized as NP Hard Problem (Tate and Smith 1995; Kochhar et al. 1998). The objective of site layout is to position temporary facilities both geographically and at the correct time such that the construction work can be served satisfactorily in order to minimize costs and improved safety and working environment.

A set of facilities needs to be located on the available location such that the total cost of construction and interactive cost due to facility layout constraints will be minimize. There are several condition become constraints in this problem, which are:

- $m$ facilities are to be positioned on a site.
- $n$ locations are available for each facility to position, $n \geq m$.
- For each assignment of a facility to a candidate location, there are different set-up and removal costs. Consequently, different assignments will mean different operational costs.
- There are adjacency constraints which dictate that certain facilities must be adjacent to other facilities.

This problem being modeled as a Quadratic Assignment Problem (QAP) and the
formulation is adapted from Yeh (1995).

\[
\begin{align*}
\text{Min } F &= \sum_{x} \sum_{i} \delta_{xi} C_{xi} + \sum_{x} \sum_{i} \sum_{j} \delta_{xi} \delta_{yj} A_{ij} D_{xy} \\
\text{s.t. } \delta_{yj} &= 0 \text{ if } \delta_{xi} = 1 \text{ and } y \neq x \\
\delta_{xj} &= 1 \text{ if } \delta_{xi} = 1 \text{ and } j \neq i
\end{align*}
\]

Where \( F \) is the cost function; \( \delta_{xi} \) is the permutation matrix variable (=1 if facility \( x \) is assigned to location \( i \)); \( C_{xi} \) is the construction cost of assigning facility \( x \) to location \( i \); \( A_{ij} = 1 \) if site \( i \) is neighboring to location \( j \); \( D_{xy} \) is the interactive cost of assigning facility \( x \) on the site neighboring facility \( y \).

3. **Overview Particle Swarm Optimization**

Particle swarm optimization (PSO) is a population based stochastic optimization technique that inspired by social behavior of bird flocking or fish schooling to a promising position for certain objectives (Kennedy and Eberhart, 1995).

The position of a particle can be used to represent a candidate solution for the problem at hand. A swarm of particles with randomly initialized positions would fly toward the optimal position along a path that is iteratively updated based on the current best position of each particle i.e., *local best* and the best position of the whole swarm i.e., *global best* (Zhang and Wang, 2008). The particle-updating mechanism is formulated as

\[
\begin{align*}
V^p(t) &= w(t) V^p(t-1) + c_1 r_1 \left( L X^p(t) - X^p(t-1) \right) + c_2 r_2 \left( -G X - X^p(t-1) \right) \\
X^p(t) &= V^p(t) + X^p(t-1)
\end{align*}
\]

Where \( X^p(t) = \{ x_1^p(t), x_2^p(t), ..., x_L^p(t) \} \) denotes the L-dimension position for the \( p \)th particle in the \( t \)th iteration, whereas \( V^p(t) = \{ v_1^p(t), v_2^p(t), ..., v_L^p(t) \} \) denotes the total number of particles in a swarm, called population size; \( t \) - 1, 2, ..., \( T \), and \( T \) denotes the total iteration limit, \( L X^p(t) = \{ l x_1^p(t), l x_2^p(t), ..., l x_L^p(t) \} \) represents the local best path for the \( p \)th particles, whereas \( G X^p(t) = \{ g x_1^p(t), g x_2^p(t), ..., g x_L^p(t) \} \) represents the global best solutions.; \( c_1 \) and \( c_2 \) are the learning factors which is positive constants, and \( r_1 \) and \( r_2 \) are the random numbers between 0 and 1; \( w(t) \) is the inertia weight used to control the impact of the previous velocities on the current one. This will cause the tradeoff between global and local experiences. \( V^p(t) \) determines a particle’s velocity.
based on its previous velocity and the distance from its current position to the better among its local best and the global best. $X^P(t)$ determines a particle’s position based on its local and global best positions.

4. **Solving Construction Site Layout Problem with Adaptive Particle Swarm Optimization**

We implemented standard PSO for the above problem and analyzed the objective function. In our standard PSO implementation we have tried to maintain the diversity with craziness concept that adopted from (Kennedy and Eberhart 1995). We found that all particles move to the same local optima and the velocities are all decayed to zero. The minimum cost after 5 runs for the standard PSO implementation are show in Table 1 below. Therefore, we decided to add cross-mutate part from genetic algorithm before updating particle’s best and global best. Here we called this algorithm as *Adaptive Particle Swarm Optimization (APSO)*. The difference with the standard PSO are:

- Change particle’s location representation
- Handling the constraints.
- Calculating Fitness value.
- Craziness added for maintain diversity in assigning velocity value.
- Change in parameter value $c_1$ and $c_2$.
- Change in value of constriction factor $k$ with associated formula.
- Added damping limit for velocity.

<table>
<thead>
<tr>
<th>Run</th>
<th>1st</th>
<th>2nd</th>
<th>3rd</th>
<th>4th</th>
<th>5th</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimum Cost</td>
<td>103</td>
<td>102</td>
<td>92</td>
<td>109</td>
<td>93</td>
</tr>
</tbody>
</table>

4.1. **Algorithmic Design for Adaptive Particle Swarm Optimization (APSO)**

In this section the representation of particle, tuning parameters, steps of the algorithm and constraint handling are presented.

4. 1. 1 **Representation of particle**

For a problem of $m$ facilities that are labeled as $A$, $B$, $C$, ..., a sequence $S$ can be established which contains a permutation of all the labels. $S$ can be interpreted as: assigning facility $S[i]$ to site $i$, where $i \in [1,m]$. Different sequences mean different
layout solutions and any manipulation of the sequence $S$ will correspond to a new layout. The number of particle’s location is adjusted with the number of location and facility in the case problem. Since the number of location and facility is the same that is 12, the number of particle’s location is set to 12 bits.

**Table 2. Particles Location Illustration**

<table>
<thead>
<tr>
<th>$i$</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S[i]$</td>
<td>5</td>
<td>3</td>
<td>4</td>
<td>1</td>
<td>9</td>
<td>10</td>
<td>7</td>
<td>11</td>
<td>12</td>
<td>2</td>
<td>6</td>
<td>8</td>
</tr>
</tbody>
</table>

$S[1] = 5 \rightarrow$ assigning facility 5 to site 1, $S[2] = 3 \rightarrow$ assigning facility 3 to site 2

### 4.1.2 Tuning parameters.

Based on some observation through trial and error during parameters testing, we determine the value of tuning parameters as follows: population size = 50, inertia weight = $(\text{maxGen}-i)/\text{maxGen}$, where in every looping, the inertia weight is always change and reduce, and maxGen = maximum number of generation/max iterations (set to 100), $i$ = generation $i^{th}$. For the acceleration constants $c_1$ and $c_2$ are set to 2.05, the velocity is calculate using $\bar{v}_i(t+1) = k \times [\bar{v}_i(t) + r_1c_1(\bar{x}_{pBest} - \bar{x}_i(t)) + r_2c_2(\bar{x}_{gBest} - \bar{x}_i(t))]$ .

Constriction Factor $k = \frac{2}{2-\varphi - \sqrt{\varphi^2 - 4\varphi}}$ where $\varphi = c_1 + c_2$, $\varphi > 4$ and for the Cross-mutate rate, we choose to applied to half of the population size with 0.7 cross mutate rate.

### 4.1.3 Steps of the algorithm & Constraint Handling

<table>
<thead>
<tr>
<th>Pseudo-code</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Program begins</strong></td>
</tr>
<tr>
<td>FOR each particle</td>
</tr>
<tr>
<td>Initialize particle randomly</td>
</tr>
<tr>
<td>FOR $i=1$ until maximum iteration</td>
</tr>
<tr>
<td>Calculate every particle velocity</td>
</tr>
<tr>
<td>Update every particles location</td>
</tr>
<tr>
<td>Calculate fitness value $F$ in every particles</td>
</tr>
<tr>
<td>FOR $cm=1$ until half of population size</td>
</tr>
</tbody>
</table>
Reproduce $p_{cm}$ using cross-mutate operation

END

Do for every particles :
  IF a particle's $F$ is better than pbest
  Update particle best(pbest) and pbest location for that particle
END

End Do

Calculate fitness value $F$ in every particles

Choose the particle $i^*$ with the best fitness

$gBest = F(i^*)$

$gBestLoc = X(i^*)$

END

Return gBest

end

**Figure 1.** APSO Pseudocode

### 4.1.4 Constraint Handling

1. **Neighboring Facility Constraint**
   
   This constraint is handled by putting a high cost in calculating fitness. In this problem, the lower the fitness value, the better the particle. Facility 1 or 2 should not be placed in neighbor of facility 10 or 11.

2. **Permutation Value Constraint for APSO**

   Permutation constraint handling in here was adopted from (Hu et al., 2003). Here the permutation was changed with a random rate defined by their velocities. In standard PSO, the velocity is added to the particle on each dimension to update the particle, thus it is a distance measure. If the velocity is larger, the particle may explore more distant areas. Similarly, the new velocity in the permutation scenario represents the possibility that the particle changes. If the velocity is larger, the particle is more likely to change to a new permutation sequence. The velocity update formula remains the same, while the particle update process is changed.
5. **Numerical Simulation**

5.1. **Numerical Example**

In our experiments, a numerical example is taken from Yeh (1995) is used to demonstrate the adaptive particle swarm formulation. We have 12 facilities to be located to 12 available sites such that the total cost of construction and interactive cost due to facility layout constraints will be minimized. The 12 facilities that need to be position in the 12 sites respectively: Reinforcing steel shop 1 (R1), Reinforcing steel shop 2 (R2), Carpentry shop 1 (C1), Carpentry shop 2 (C2), Falsework shop 1 (F1), Falsework shop 2 (F2), Concrete batch plant 1 (B1), Concrete batch plant 2 (B2), Job office (JO), Labor residence (LR), Electricity equipment and water-supply shop (E), Warehouse (W)

![Diagram](image)

**Figure 2.** The example site layout problem (Yeh, 1995)

Construction cost matrix (C) are shown in Tables 3. For site neighboring index matrix (A) and interactive cost matrix (D) data, please refer to Mawdesley and Al-Jibouri (2003). In their formulation, the site neighboring index matrix is slightly different to that proposed by Yeh (1995) since on the site, location 1 is not adjacent to location 12 as suggested in the original. Here we will use this formulation for the site neighboring index.

<table>
<thead>
<tr>
<th>Table 3. Construction Cost Matrix (C)</th>
</tr>
</thead>
</table>

7
5.2. Experiments Result

Computational experiments are presented to justify the APSO algorithm for solving the construction site layout problem. To investigating the efficiency of our proposed algorithm when adopting the adaptive approach, we will compare the performance of the APSO-based method with annealed neural network-based method from Yeh (1995).

A package was written in MATLAB to solve the problem under consideration. The package runs under Windows XP SP. 3 environment. A generation of the problem with a population of 100 took approximately 0.58s in average to run on a P 2.2 processor with 1 GB of RAM. This timing was taken by averaging and no attempt has been made to optimize the programming. Figure 3 show the result of PSO for 100 generations, here the APSO can obtain the minimum cost 90 at 22 generations, while the best solution obtained by Yeh (1995) had minimum cost of 93.0. Average objective value was 90.6 which is lower than obtained by Yeh. Table 5 shows the optimal positioning of the facility correspondent to its location obtains by APSO. The locations of the facilities obtained by APSO algorithm and by Yeh (1995) are compared in Table 6.

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1</td>
<td>35</td>
<td>35</td>
<td>30</td>
<td>30</td>
<td>35</td>
<td>15</td>
<td>10</td>
<td>15</td>
<td>6</td>
<td>6</td>
<td>7</td>
<td>10</td>
</tr>
<tr>
<td>R2</td>
<td>35</td>
<td>30</td>
<td>9</td>
<td>9</td>
<td>13</td>
<td>30</td>
<td>30</td>
<td>35</td>
<td>15</td>
<td>18</td>
<td>12</td>
<td>7</td>
</tr>
<tr>
<td>C1</td>
<td>18</td>
<td>15</td>
<td>15</td>
<td>15</td>
<td>8</td>
<td>14</td>
<td>10</td>
<td>8</td>
<td>10</td>
<td>15</td>
<td>15</td>
<td>15</td>
</tr>
<tr>
<td>C2</td>
<td>13</td>
<td>7</td>
<td>12</td>
<td>18</td>
<td>18</td>
<td>15</td>
<td>15</td>
<td>15</td>
<td>8</td>
<td>8</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>F1</td>
<td>18</td>
<td>15</td>
<td>15</td>
<td>20</td>
<td>15</td>
<td>8</td>
<td>10</td>
<td>8</td>
<td>8</td>
<td>7</td>
<td>15</td>
<td>15</td>
</tr>
<tr>
<td>F2</td>
<td>14</td>
<td>8</td>
<td>10</td>
<td>17</td>
<td>12</td>
<td>15</td>
<td>15</td>
<td>15</td>
<td>8</td>
<td>7</td>
<td>9</td>
<td></td>
</tr>
<tr>
<td>B1</td>
<td>32</td>
<td>35</td>
<td>15</td>
<td>15</td>
<td>15</td>
<td>10</td>
<td>9</td>
<td>13</td>
<td>7</td>
<td>10</td>
<td>15</td>
<td>15</td>
</tr>
<tr>
<td>B2</td>
<td>31</td>
<td>30</td>
<td>9</td>
<td>8</td>
<td>15</td>
<td>18</td>
<td>15</td>
<td>16</td>
<td>15</td>
<td>15</td>
<td>15</td>
<td>15</td>
</tr>
<tr>
<td>JO</td>
<td>39</td>
<td>35</td>
<td>13</td>
<td>8</td>
<td>8</td>
<td>15</td>
<td>18</td>
<td>15</td>
<td>8</td>
<td>18</td>
<td>9</td>
<td>18</td>
</tr>
<tr>
<td>LR</td>
<td>18</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>15</td>
<td>10</td>
<td>15</td>
<td>13</td>
<td>15</td>
<td>15</td>
<td>15</td>
<td>15</td>
</tr>
<tr>
<td>E</td>
<td>7</td>
<td>10</td>
<td>8</td>
<td>19</td>
<td>15</td>
<td>10</td>
<td>10</td>
<td>8</td>
<td>15</td>
<td>10</td>
<td>6</td>
<td>15</td>
</tr>
<tr>
<td>W</td>
<td>9</td>
<td>10</td>
<td>6</td>
<td>7</td>
<td>7</td>
<td>15</td>
<td>15</td>
<td>18</td>
<td>15</td>
<td>15</td>
<td>15</td>
<td>12</td>
</tr>
</tbody>
</table>
Table 4. Average time and minimum cost obtained by APSO

<table>
<thead>
<tr>
<th>Run</th>
<th>1st</th>
<th>2nd</th>
<th>3rd</th>
<th>4th</th>
<th>5th</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimum cost</td>
<td>91</td>
<td>90</td>
<td>92</td>
<td>90</td>
<td>90</td>
</tr>
<tr>
<td>CPU time</td>
<td>0.580377</td>
<td>0.562150</td>
<td>0.577851</td>
<td>0.595490</td>
<td>0.589239</td>
</tr>
</tbody>
</table>

Table 5. Positioning of the facilities to the corresponding sites from the optimal solution by APSO

<table>
<thead>
<tr>
<th>Fac</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td>Loc</td>
<td>10</td>
<td>12</td>
<td>9</td>
<td>2</td>
<td>8</td>
<td>11</td>
<td>7</td>
<td>4</td>
<td>5</td>
<td>3</td>
<td>1</td>
<td>6</td>
</tr>
</tbody>
</table>

Table 6. Comparison of APSO and Yeh (1995)

<table>
<thead>
<tr>
<th>Objective</th>
<th>R1</th>
<th>R2</th>
<th>C1</th>
<th>C2</th>
<th>F1</th>
<th>F2</th>
<th>B1</th>
<th>B2</th>
<th>JO</th>
<th>LR</th>
<th>E</th>
<th>WH</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yeh (1995)</td>
<td>93.0</td>
<td>12</td>
<td>9</td>
<td>8</td>
<td>1</td>
<td>7</td>
<td>3</td>
<td>4</td>
<td>11</td>
<td>6</td>
<td>2</td>
<td>10</td>
</tr>
<tr>
<td>APSO</td>
<td>90.0</td>
<td>10</td>
<td>12</td>
<td>9</td>
<td>2</td>
<td>8</td>
<td>11</td>
<td>7</td>
<td>4</td>
<td>5</td>
<td>3</td>
<td>1</td>
</tr>
</tbody>
</table>

6. Conclusion

Here an adaptive particle swarm optimization is presented to tackle the local optima.
problem that used face by the ordinary PSO. The main idea is to maintain the diversity with craziness concept and then added the cross-mutate part from genetic algorithm before updating particle’s best and global best. An example problem where 12 facilities need to be positioned to 12 sites taken from (Yeh 1995) is use to show the performance of the proposed algorithm. Experiment results show that APSO converges after twenty two iterations with minimum construction cost 90 and average for best objective is 90.6 for five times runs. Hence, our proposed algorithms perform better than the previous proposed method and it is efficient in solving the construction site layout problem.

REFERENCES


of Computing In Civil Engineering, ASCE, 9:3, 201–208.