Simplified swarm optimization for optimal deployment of fog computing system of industry 4.0 smart factory

Wei-Chang Yeh\textsuperscript{1}, Chyh-Ming Lai\textsuperscript{2}, and Jr-Yu Tsai\textsuperscript{1,*}

\textsuperscript{1}Integration and Collaboration Laboratory, Department of Industrial Engineering and Engineering Management, National Tsing Hua University, Hsinchu 300, Taiwan
\textsuperscript{2}The Institute of Resources Management and Decision Science, Management College, National Defense University, Taipei 112, Taiwan

* Corresponding author: heal.tsai@gmail.com

Abstract. Fog computing is an emerging technology that can reduce the load on cloud system and decentralize computing resource, thus increasing response speed and reducing time delay. More and More environments in order to achieve intelligence, collect massive amounts of data through IOT devices. Considering deploying fog computing system in these environments can make system faster and more robust. This work creates a simulated factory consisting of a cloud center, gateways, fog devices, edge devices, and different types of sensors. We build an integer programming model and apply metaheuristic method for determining the layout of above devices except of cloud center and all sensors so that can find the cost minimization deployment of the smart factory.

1. Introduction
Benefiting from the advancement of cloud computing technology and hardware, the Internet of Things (IoT) is widely used in various products and environments in today's society. The IoTs collects a large amount of data through sensing devices such as radio frequency identification (RFID), sensors and mobile internet device, etc., and processes the huge amount of data through cloud computing system. However, today’s IoT application environments are increasingly sensitive to delay tolerance, such as smart factories in fierce industry competition, in order to maximize profits, need to have the ability to flexibly produce to meet the needs of customization, and also when the machine occurs faults, need to have ability to repair quickly to minimize downtime.

Generally speaking, the cloud computing system collects data on the site by the sensing device, uploads the cloud center and downloads it back to the decision marker. When the amount of data exceeds the cloud system computing load, waiting for data uploading and computing will inevitably occur, and the data that must be processed immediately will lose its value. For that, in 2012, Cisco proposed a concept named fog computing, as the extension of original cloud computing system[1]. This technology creates a buffer for cloud computing and decentralized resource allocation to alleviate its computing load.

Since the concept of fog computing was proposed, the deployment of fog computing systems or wireless sensing devices has attracted the attention of many scholars. Past research has minimized the number of settings [2] [3], minimized response time [4], and minimized setup costs [5] [6], etc. The application environment included smart factories, smart cities, car networking, smart healthcare, and intelligent logistics centers. Lin and Yang (2018) [5] first proposed a model that considers both
response time and setup cost, and separates the wireless access device (WAP) as the edge device layer. This research not only integrated the results of past research, but also provided a new hierarchical structure to make the division of labor clearer. However, in order to simplify the problem, the sensing device is limited to the Automated Guided Vehicle (AGV), which is unreasonable in practical use.

In the wave of Industry 4.0, Smart Factory has become a suitable scenario for considering the deployment of fog computing systems for the following reasons:

1) The factory sets up various types of sensing devices and automated machines to achieve intelligent manufacturing, and increases the amount of data generated on site.

2) Accelerated response speed in factories often increases the profit or reduces the loss. For example, the faster the machine parameters are adjusted, the faster the material can enter the production line and the lead time is shortened.

This study constructs a reasonable cost minimization fog computing system deployment model about the virtual smart factory environment. The model belongs to the single target optimization facility location problem which has been proved to be NP-Hard [7], so many studies have used evolutionary algorithms to optimize the solution. We apply Simplified Swarm Optimization (SSO) to deal with the proposed problem, since SSO performs well on discrete problems [8] [9]; and compare it with the famous discrete algorithm-gene algorithm (GA). The expected contributions of this study are as follows:

1) Construct a more appropriate integer calculation model for the fog calculation system.

2) Use a simple and effective heuristic algorithm, SSO, to solve this novel problem and provide a different method reference for similar research in the future.

2. Literature review

2.1. Facility location problem

The problem of deployment of fog computing system is the deformation of problem of hierarchical facility location problem (HFLP) [10], and considers the constraints about device load and data processing capacity additionally. Guo, Lin, Li, He, and Li [6] proposed two different integer programming models in 2016 on the deployment of fog computing systems to transportation networks. In one of the models called DRF, Road-Side Unit (RSU) can only transmit and receive data. Another model is CRF, RSU is not only a wireless access point (WAP), but also has the functions of calculation and communication.

In the same year, Xu, Liang, Xu, Jia and Guo [4] proposed an integer programming model of deployment of fog computing system considering the data length, transmission rate and device capacity limitation in order to minimize the response time, and using four different algorithms to solve. In 2018, Lin and Yang extended Guo et al’s model architecture, and separated WAP as an edge device layer. Combined with its CRF model and Xu et al’s concept of response time as constraints, a novel and more comprehensive mathematical model was developed.

2.2. Simplified swarm optimization

SSO is a heuristic algorithm proposed by Yeh [11] to solve the shortage of PSO in discrete problems. Using the concept of step function and target, a simple and fast variable optimization method is established. The update mechanism SSO is as follows:

\[
x_{ij}^t = \begin{cases} 
g_i & \text{if } r_{ij}^t \in (0, C_g = c_g) \\
g_{ij}^{-1} & \text{if } r_{ij}^t \in [C_g, C_p = C_g + c_p) \\
g_{ij} & \text{if } r_{ij}^t \in [C_p, C_e = C_p + c_e) \\
x & \text{if } r_{ij}^t \in (C_e, 1]
\end{cases}
\]  

(1)

\(c_g, c_p, \) and \(c_e\) are defined parameters, and the sum of \(c_g, c_p,\) and \(c_e\) is 1. Usually \(c_g\) is the largest, \(c_p\) is the second, and \(c_e\) is the smallest, that is, a maximum probability interval is set to extract information from global best solution, and retains the ability of global search simultaneously. \(x_{ij}^t\) represents a set of
solutions, $i$ and $j$ denote the $j$th variable of $i$th row and $t$ denotes the number of iterations. $\rho$ is a random number from standard uniform distribution, used to determine the update method. If $\rho$ is in the interval $(0, C_g=c_g)$, then the $j$th variable of the global best solution is taken; if in interval $[C_g, C_p)$, take the $j$th variable of the best solution of the previous generation; if in interval $[C_p, C_w)$, follow the previous generation variable value; if in interval $[C_w, 1)$, then randomly generates variable value.

3. Equations and mathematics

3.1. Preliminary work
This work assumes a virtual textile smart factory like Fig. 1, based on the actual layout of the traditional textile factory.

![Figure 1. Deployment of textile factory.](image)

![Figure 2. Deployment of the simulated smart factory.](image)
In general, the process from fabrics to garments goes through relaxing fabrics, sewing, packaging and inventory, so we divided the factory into three main parts, namely fabric relaxation area, sewing area, and warehouse. In each of these three parts, different sensing devices that should be set. Such as fabric relaxation area should have temperature, humidity and object sensing devices. Sewing area only requires object sensing devices. Sensing devices required by warehouse are same as fabric relaxation area. In addition, AGVs are required in the middle of each area for delivery.

First, fix the positions of sensing devices and cloud center, and then randomly arrange the potential positions of edge devices, fog devices, and the gateway as shown in Fig. 2. The black square is the center of the cloud; the blue square is the potential position of the fog device; the red square is the potential position of the edge device; the red dot is the temperature sensing device; the cyan dot is the humidity sensing device; the blue dot is the object class Sensing device; the green dot is the position of the AGV.

3.2. Problem model

Before we build the model, we must define the notations given in Table 1.

Table 1. Notations.

| Decision variable | Definition | Decision variable | Definition |
|-------------------|------------|-------------------|------------|
| $x_{ij}$          | A binary variable deciding if a link between nodes i and j exists. | $f_b$      | A binary variable deciding if the potential site for fog device b is selected to place a fog device. |
| $g_a$             | A binary variable deciding if the potential site for gateway a is selected to place a gateway. | $e_c$      | A binary variable deciding if the potential site for edge device c is selected to place an edge device. |

| Parameter | Definition | Parameter | Definition |
|-----------|------------|-----------|------------|
| C         | Index of cloud center. | $d_{ij}$ | Distance between nodes i and j. |
| $\Omega_c$ | Set of potential sites for gateways. | $L_{ro}$ | Maximum latency time of linking edge device c to the object sensing device o. |
| $\Omega_f$ | Set of potential sites for fog devices. | $L_{od}$ | Maximum latency time of linking edge device c to the AGV d. |
| $\Omega_e$ | Set of potential sites for edge devices. | $L_{ec}$ | Maximum latency time of linking fog device b to edge device c. |
| $\Omega_{ST}$ | Set of potential sites for temperature sensing devices. | $L_{ek}$ | Maximum latency time of linking gateway a to fog device b. |
| $\Omega_{SH}$ | Set of potential sites for humid sensing devices. | $DL_F$ | The data length for a gateway from a fog device. |
| $\Omega_{SO}$ | Set of potential sites for object sensing devices. | $DL_E$ | The data length for a fog device from an edge device. |
| $\Omega_A$ | Set of potential sites for AGV. | $DL_{AG}$ | The data length for an edge device from an object sensing device. |
| $c_f$ | The price of one unit length of fiber. | $DL_A$ | The data length for an edge device from an AGV. |
| $c_g$ | Cost of installing a gateway. | $\gamma_f$ | The data rate from a fog device to a gateway. |
| $c_f$ | Cost of installing a fog device. | $\gamma_E$ | The data rate from an edge device to a fog device. |
| $c_E$ | Cost of installing an edge. | $\gamma_{ST}$ | The data rate from a temperature sensing device to an edge device. |
| $H^E_a$ | The maximal demand that | $\gamma_A$ | The data rate from an AGV to an edge device |


The maximal demand that fog device $b$ can fulfill.

$H_b^f$ 

Diameter of the coverage range of an edge device.

$R_e$ 

Demand of a temperature sensing device.

$r_{ST}$ 

Demand of a humd sensing device.

$r_{SH}$ 

Demand of an object sensing device.

$r_{SO}$ 

Demand of an AGV.

$r_A$ 

The maximum number of fog devices that a gateway can accommodate.

$N_{G}$ 

The maximum number of edge devices that a fog device can accommodate.

$N_{f}$ 

The maximum number of sensing devices which are covered by an edge device.

$N_{E}$ 


---

| Variable | Definition |
|----------|------------|
| $\alpha (P_i)$ | Demand at potential site $P_i$. |

The model for deployment of fog computing systems is an integer programming model as follow:

\[
\text{Min} \quad \text{Cost} = c_r \sum_{y \in \Omega_y} g_y + c_f \sum_{y \in \Omega_y} f_y + c_e \sum_{y \in \Omega_y} e_y + c_i \sum_{(i,j) \in \Omega_i, (a,b) \in \Omega_a, (c,d) \in \Omega_c} x_{cd} d_y 
\]  

(2)

Subject to (3)-(29)

\[
\sum_{y \in \Omega_y} g_y = 1, \forall y \in \Omega_y 
\]  

(3)

\[
\sum_{y \in \Omega_y} f_y = 1, \forall y \in \Omega_y 
\]  

(4)

\[
\sum_{y \in \Omega_y} e_y = 1, \forall y \in \Omega_y 
\]  

(5)

\[
\sum_{y \in \Omega_y} x_{cd} = 1, \forall y \in \Omega_y 
\]  

(6)

\[
x_e, x_a, x_b, x_c \leq 1, \forall e, \forall a, \forall b, \forall c \in \Omega_e 
\]  

(7)

\[
x_e, x_a, x_b, x_c \leq 1, \forall e, \forall a, \forall b, \forall c \in \Omega_e 
\]  

(8)

\[
\sum_{y \in \Omega_y} x_{cd} = 1, \forall y \in \Omega_y 
\]  

(9)

\[
f_y \leq f_j, \forall b, \forall e \in \Omega_y 
\]  

(10)

\[
x_e \leq x_a, \forall b, \forall e \in \Omega_y 
\]  

(11)

\[
x_e \leq x_a, \forall b, \forall e \in \Omega_y 
\]  

(12)

\[
x_e \leq x_a, \forall b, \forall e \in \Omega_y 
\]  

(13)

\[
x_e \leq x_a, \forall b, \forall e \in \Omega_y 
\]  

(14)

\[
x_e \leq x_a, \forall b, \forall e \in \Omega_y 
\]  

(15)

\[
\sum_{y \in \Omega_y} r_{ab} x_d = 1, \forall y \in \Omega_y 
\]  

(16)

\[
\sum_{y \in \Omega_y} r_{ab} x_d = 1, \forall y \in \Omega_y 
\]  

(17)

\[
\sum_{y \in \Omega_y} r_{ab} x_d = 1, \forall y \in \Omega_y 
\]  

(18)

\[
\sum_{y \in \Omega_y} r_{ab} x_d = 1, \forall y \in \Omega_y 
\]  

(19)

\[
\sum_{y \in \Omega_y} r_{ab} x_d = 1, \forall y \in \Omega_y 
\]  

(20)

\[
\sum_{y \in \Omega_y} r_{ab} x_d = 1, \forall y \in \Omega_y 
\]  

(21)

\[
DL_{SO} / \sum_{y \in \Omega_y} x_{y} \leq L_{SO}, \forall y \in \Omega_y 
\]  

(22)

\[
DL_{ST} / \sum_{y \in \Omega_y} x_{y} \leq L_{ST}, \forall y \in \Omega_y 
\]  

(23)

\[
DL_{HT} / \sum_{y \in \Omega_y} x_{y} \leq L_{HT}, \forall y \in \Omega_y 
\]  

(24)

\[
DL_{SH} / \sum_{y \in \Omega_y} x_{y} \leq L_{SH}, \forall y \in \Omega_y 
\]  

(25)
where \( x_i, x_j, x_k, x_m, x_n, x_p \leq R_i/2 \)

\[
\sum_{i \in \Omega_c} x_i + \sum_{j \in \Omega_d} x_j + \sum_{k \in \Omega_e} x_k \leq N_c, \forall c \in \Omega_c
\]

\[
\sum_{i \in \Omega_d} x_i \leq N_d, \forall d \in \Omega_d
\]

\[
\sum_{i \in \Omega_e} x_i \leq N_e, \forall e \in \Omega_e
\]

Constraints (3) to (15) are mainly described the limits on the number of links, (16) to (21) are limits for data processing, (22) to (25) are limits for data transmission delay, (26) is limit for coverage of edge devices, and (27) to (29) are limits for capacity of each layer device. It is worth mentioning that we consider that the sensing data of temperature and humidity is not urgent, so the limitation of such sensing devices is not in transmission delay.

4. Proposed method

4.1. Process of simplified swarm optimization

In this section, we introduce SSO how to apply for cost minimization deployment problem. First we have to decide value of \( c_g, c_p, c_w \), \( N_{run}, N_{gen}, N_{sol} \) and \( N_{var} \). The above \( c_g, c_p, c_w \) have been mentioned before, \( N_{run} \) represents the number of experiments, \( N_{gen} \) represents the number of iterations, \( N_{sol} \) represents the number of solutions entering the solution space, and \( N_{var} \) represents the number of variables in a solution. The entire SSO process is outlined in Table 2.

**Table 2. Process of SSO.**

| Step | Description |
|------|-------------|
| 1    | Generate initial solutions \( X^{0\text{th}} \) and calculate the value of fitness function \( F(X^{0\text{th}}), i=1,2,...,N_{sol} \). |
| 2    | Let \( g_{\text{Best}} \) be the solution with the smallest value in \( F(X^{0\text{th}}), i=1,2,...,N_{sol} \), and then let \( t=1 \). |
| 3    | Update from the first to \( N_{var} \) variable of \( X_i, i=1,2,...,N_{sol} \). |
| 4    | Evolve \( X^{(t+1)} \) and recalculate the value of fitness function \( F(X^{(t+1)}), i=1,2,...,N_{sol} \). |
| 5    | If \( F(X^{(t+1)})<F(X_i) \), let \( X_i=X^{(t+1)} \); if not, then \( X_i \) remains unchanged and skips to step 7. |
| 6    | If \( F(X^{(t+1)})<F(g_{\text{Best}}) \), let \( g_{\text{Best}} = X^{(t+1)} \). |
| 7    | If \( i<N_{sol} \), let \( i=i+1 \) and return to step 4. |
| 8    | If \( t=N_{gen} \), the program terminates; otherwise \( t=t+1 \) and return to step 3. |

4.2. Encoding method

The encoding method presents a solution with a set of 0, 1, where 0 indicates that the device at this potential location is closed, and conversely 1 indicates that the device at this potential location is opened. The first part of the code is the list of potential positions of the gate, the middle part is the list of potential positions of the fog computing device, and the last part is the list of potential positions of the edge device, as shown in Fig. 3.
4.3. Fitness function calculation

The fitness function is an important indicator for evaluating the metaheuristic algorithm. Based on the integer programming model in Chapter 3, we convert (2) into (30):

\[
\text{Min} \quad Cost = c_g \sum_{a \in \Omega} n_a + c_f \sum_{f \in \Omega} f_n + c_e \sum_{e \in \Omega} e_n + c_c \sum_{i,j \in \Gamma_{a,p}, a_p \in \Omega} x_{a_p,i,j} d_{i,j} + \text{Penalty} \times (n_{\text{link}} + n_{\text{demand}} + n_{\text{latency}} + n_{\text{cover}} + n_{\text{capacity}})
\]  

(30)

\(n_{\text{link}}, n_{\text{demand}}, n_{\text{latency}}, n_{\text{cover}}\) and \(n_{\text{capacity}}\) represent the number of violating corresponding constraints. The fitness value is calculated as shown in the following Table 3.

Table 3. Process of fitness value calculation.

| Step | Description |
|------|-------------|
| 1:   | Randomly generate a list of random 0, 1, and set the gateways, fog devices and edge devices must have at least one 1, if not, then \(n_{\text{id}}\) increases 1. |
| 2:   | Find the nearest device connected to the upper level. |
| 3:   | Calculate capacity and data processing capacity of gateways, fog devices and edge devices. If it is higher than the set value, the \(n_{\text{capacity}}\) and \(n_{\text{demand}}\) increase 1. |
| 4:   | Calculate the data transmission delay. If it is higher than the set value, the \(n_{\text{latency}}\) increases 1. |
| 5:   | If the distance between the sensing devices and the opened edge devices is greater than \(r_e/2\), \(n_{\text{cover}}\) increases 1. |
| 6:   | Multiply list by the corresponding cost, multiply the total distance by the cost of the fiber, and multiply \(\text{Penalty}\) by the number of violations. |
| 7:   | Add all multiplication values. |

5. Numerical implement

In this section, we implement SSO on a simulated smart factory environment. The parameters of this simulated environment are given as follow Table 4.

Table 4. Parameter setting.

| Parameter | Value |
|-----------|-------|
| \(c_f\)   | 50    |
| \(c_G\)   | 200   |
| \(c_f\)   | 100   |
| \(c_E\)   | 80    |
| \(H_{G}^c\)| 300000|
| \(H_{F}^c\)| 30000 |
| \(H_{E}^c\)| 3000  |
| \(N_G\)   | 10    |
| \(N_F\)   | 10    |
We compare the result obtained by SSO with those obtained by traditional GA. We generate three different scale of questions for experimentation. In the small scale problem, the number of \( N_{var} \) is 50(2 gateways, 10 fog devices and 38 edge devices); In the middle scale problem, the number of \( N_{var} \) is 88(3 gateways, 15 fog devices and 70 edge devices); In the big scale problem, the number of \( N_{var} \) is 156(6 gateways, 30 fog devices and 120 edge devices). \( N_{run} \), \( N_{gen} \), \( N_{sol} \) of SSO and GA are the same, the values are 20, 100, 100 respectively. SSO parameter \( C_g \) is set to 0.45, \( C_p \) is set to 0.75, \( C_w \) is set to 0.9, and GA parameter crossover rate is set to 0.8, mutation rate is set to 0.03.

Table 5. The results of SSO and GA.

| Scale          | SSO     | GA     |
|---------------|---------|---------|
| Small Scale   |         |         |
| Fitness value | Avg.    | Std.    | Avg.   | Std.   |
|               | 44803.60| 1271.67 | 50053.25| 4095.44|
| time          | 43.81   | 0.50    | 130.49 | 1.69   |
| Best          | 42546.04|         | 44798.79|       |
| Middle Scale  |         |         |
| Fitness value | Avg.    | Std.    | Avg.   | Std.   |
|               | 53248.67| 1766.85 | 57833.80| 4485.22|
| time          | 82.22   | 2.52    | 177.89 | 3.60   |
| Best          | 48033.47|         | 51393.20|       |
| Big Scale     |         |         |
| Fitness value | Avg.    | Std.    | Avg.   | Std.   |
|               | 78120.78| 1796.258| 88149.45| 4367.68|
| time          | 127.73  | 2.72    | 292.99 | 3.73   |
| Best          | 74703.55|         | 79975.35|       |

From the Table 5, we can see that SSO can find better solution than GA. We think this is because SSO is a kind of all-variable update method, and the ability to escape the local optimal solution is prominent. However, GA in this encoding method and a highly restricted problem model, the crossover often causes infeasible solution. In addition, the update mechanism of GA does not refer to
pBest and gBest, so it depends on the initial solution. SSO is superior to GA in the quality of solution and time, so we regard that SSO is effective in solving such problems. Fig. 4 is the middle-scale cost-minimized fog computing system layout diagram obtained from SSO.

![Deployment of middle-scale problem(SSO).](image)

**Figure 4.** Deployment of middle-scale problem(SSO).

6. Conclusion
This work continues the structure and model proposed by the previous scholars and is slightly modified to better match the situation at the smart factory deployment. We additionally consider the diversity of sensing devices and implement it in the integer programming model. Further applying SSO to the deployment problem of fog computing system also yield good results.

In the future, this work can be extended in the following directions:
1) Diverse application environment
The rise of various IoT environments, the demand for decentralized computing systems is also growing.
2) Flexible model planning
The deployment of fog computing systems in various environments also requires consideration of different restrictions.
3) Encoding methods and algorithms
Finding a more efficient algorithm to solve this problem is also part of the interest of this research. Under this highly restricted model, constructing a better coding method is also a problem that can be studied.

7. References
[1] F. Bonomi, R. Milito, J. Zhu, S. Addepalli, In Proc. 1st MCC Workshop on Mobile Cloud Computing, Fog computing and its role in the internet of things, 13-16, (2012).
[2] J. Lee, T. Kwon, J. Song, IEEE Trans. Ind. Electron, Group connectivity model for industrial wireless sensor networks, 57, 1835-1844, (2010).
[3] C.C. Lin, D.J. Deng, Z.Y. Chen, K.C. Chen, IEEE Commun. Mag., Key design of driving industry 4.0: Joint energy-efficient deployment and scheduling in group-based industrial wireless sensor networks, 54, 46-52, (2016).
[4] Z. Xu, W. Liang, W. Xu, M. Jia, S. Guo, IEEE Trans. Parallel Distrib. Syst., Efficient algorithms for capacitated cloudlet placements, 27, 2866-2880, (2016).
[5] C.C. Lin, J.W. Yang, IEEE Trans. Ind. Informat., Cost-efficient Deployment of Fog Computing Systems at Logistics Centers in Industry 4.0, 14, 4603-4611, (2018).

[6] P. Guo, B. Lin, X. Li, R. He, S. Li, IEEE TrustCom, Optimal deployment and dimensioning of fog computing supported vehicular network, 2058-2062, (2016).

[7] N. Megiddo, A. Tamir, OPER. RES. LETT., On the complexity of locating linear facilities in the plane, 1, 194-197, (1982).

[8] C.M. Lai, W.C. Yeh, Y.C. Huang, Appl. Soft Comput., Entropic simplified swarm optimization for the task assignment problem, 58, 115-127, (2017)

[9] W.C. Yeh, Comput. Oper. Res., Simplified swarm optimization in disassembly sequencing problems with learning effects, 39, 2168-2177, (2012).

[10] V. Marianov, D. Serra, EUR. J. OPER. RES., Hierarchical location–allocation models for congested systems, 135, 195-208, (2001).

[11] W.C. Yeh, Expert Syst. Appl., A two-stage discrete particle swarm optimization for the problem of multiple multi-level redundancy allocation in series systems, 36, 9192-9200, (2009).