Deployment optimization for a long-distance wireless backhaul network in industrial cyber physical systems

Jintao Wang1,2, Xi Jin1, Peng Zeng1, Ming Wan1 and Changqing Xia1

Abstract
Industrial wireless networks are an important component of industrial cyber physical systems, and their transmission performance directly determines the quality of the entire system. During deployment, the nodes of an industrial wireless network can be deployed in only some specific regions due to physical environment restrictions in the factory; thus, occlusions are not always effectively circumvented and network performance is reduced. Therefore, this article focuses on the layout problem of the industrial backhaul network: a WiFi long-distance, multi-hop network. The optimization objectives were network throughput and construction cost, and the network delay was used as a constraint. For small networks, we propose a hierarchical traversal method to obtain the optimal solution, whereas for a large network, we used a hierarchical heuristic method to obtain an approximate solution, and for extremely large networks, we used a parallel interactive local search algorithm based on dynamic programming. Then, if the original network layout cannot meet the transmission demands due to traffic bursts, we propose a network bandwidth recovery method based on the Steiner tree to recover the network’s performance. Finally, the results of a simulation showed that the algorithms proposed in this article obtain an effective solution and that the heuristic algorithm requires less computing time.

Keywords
Cyber physical systems, industrial wireless network, network optimization, node deployment, heuristic algorithm

Date received: 30 June 2017; accepted: 23 October 2017

Handling Editor: Suparna De

Introduction
As wireless communication technology has developed, wireless technologies based on protocols such as WirelessHART,1 WIA-PA,2 and ISA100.11a3 have been widely applied in industrial cyber physical systems (CPS).4 However, because of restrictions to capital, technology, and equipment, among others, there may be a situation in which multiple network technologies coexist in a given period, thus forming a heterogeneous industrial wireless control network.

IEEE 802.11–based long-distance WiFi networks (WiFi-based long-distance (WiLD) multi-hop networks)5 can achieve long-distance signal transmission because the nodes incorporate high-power IEEE802.11 wireless cards and high-gain directive antennas. In addition, a single-hop link can be tens or hundreds of kilometers in length and feature high bandwidth, low cost, and wide coverage; therefore, it can provide

1Industrial Control Network and Systems Department, Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang, China
2University of Chinese Academy of Sciences, Beijing, China

Corresponding author:
Peng Zeng, Industrial Control Network and Systems Department, Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang, China.
Email: zp@sia.cn

Creative Commons CC-BY: This article is distributed under the terms of the Creative Commons Attribution 4.0 License (http://www.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 (http://www.uk.sagepub.com/aboutus/openaccess.htm).
backhaul access for the wireless network to achieve long-distance transmission of data.

This article considers the heterogeneous industrial monitoring network used in industrial CPS, as shown in Figure 1. The wireless network (such as WirelessHART, WIA-PA, or ZigBee) applied in each factory is considered a subnet of a heterogeneous network, and each subnet is connected via an industrial backhaul network through IEEE 802.11 long-distance WiFi to constitute the wide-area heterogeneous industrial monitoring network. A control command given by the general control center is transmitted to the control plant in the subnet through the backhaul network and the subnet gateway, in that order. Meanwhile, the backhaul network provides an interconnection between the industrial control network and the Internet.

However, during the practical deployment of a network, WiLD nodes can be deployed in only specific regions due to physical environment restrictions in the factory. Meanwhile, a high tower cannot be erected in some factories, and an antenna can be erected only at the top of a building or existing lamp pole. As a result, the WiLD network’s antenna fails to effectively elude some occlusions, thus hindering the network’s transmission performance. Therefore, the layout of the network should be optimized. Figure 2 shows a partial map of Fushun Petrochemical Factory, China National Petroleum Corporation, in which the circled positions are metal tank areas or pipeline areas whose occlusion cannot be directly circumvented. In this case, it is necessary to consider the proper locations and number of nodes, the angle of the antenna, and other factors to optimize the network bandwidth and guarantee network performance.

This problem can be included in the scope of facility location. It is most similar to the P-median problem in facility location but with the following differences:

1. The traditional P-median problem is used to determine the minimum sum of the weights from demand points to the facility location, whereas in the network model of this article, consideration is given not only to the weights from demand points (gateway node) to the facility (exchange board) location but also to the weights between facilities through which the traffic flow passes.
2. Only a single target, namely, the weighted distance, is considered in the traditional P-median model, whereas the model in this article targets the optimization of the overall bandwidth and system overhead to plan the deployment of network nodes.

Thus, in this article, we used a network architecture based on WiLD multi-hop networks to optimize the node layout of the industrial CPS. The optimization
objectives were network throughput and construction cost, and the network delay was used as a constraint. The main contributions of the article are as follows:

1. We propose a hierarchical P-median optimization model based on the traditional P-median model. There are two methods in this model. First, for a small network, a hierarchical traversal method based on the hierarchical P-median optimization model was adopted to obtain the optimal solution. Second, for a large network, a hierarchical heuristic method based on the hierarchical P-median optimization model was used to obtain an approximate solution.

2. To cope with the situation when the original network layout cannot meet the transmission demands due to traffic bursts, we propose a network bandwidth recovery method based on the Steiner tree (BRM-ST) to recover the network’s performance.

3. A test was carried out via simulation, and a comparison was made between the result obtained using the algorithm proposed in this article and that obtained using the traditional genetic algorithm (GA)–based heuristic method. The result showed that the proposed algorithm obtains an effective solution and that the heuristic algorithm requires less computing time.

The remainder of this article is organized as follows. We summarize the related works in section “Related work.” We formulate the problem and provide the system model in section “System model and formulation of the problem.” Following the model description, two methods based on hierarchical P-median optimization model and a parallel interactive local search algorithm based on dynamic programming (PILS-DP) are proposed in section “Solutions.” In section “Network node deployment for traffic bursts,” we propose network node deployment methods for traffic bursts. The experimental results are given in section “Experiment and analysis.” Finally, section “Conclusion” concludes the article.

Related work

Numerous studies of facility location models have been conducted by domestic and foreign scholars. Maximum Covering Location is often adopted when the decision-maker fails to satisfy all demands using the available resources (namely, fails to fully satisfy the demands under the given construction input and distance). Set Covering Location is used to obtain the lowest possible facility construction cost while satisfying all demands under a given emergency deadline. The P-median model determines the P facility locations and minimizes the weighted distance from the demand point to the facility (distance can also be expressed as traffic/transport time).

The traditional facility location problems include the p-center and p-coverage problems, which are NP-complete problems. The existing research is more focused on the p-center problem. And some approximation algorithms such as branch-and-bound, branch-and-price, Lagrangian relaxation, integral programming, linear programming relaxation, and randomized rounding are widely used to solve the p-center problem and no capacity limits facility location problems. Besides, the GA, neural network, and simulated annealing are used to solve the large-scale p-center problem.

The current wireless network connection recovery methods are roughly divided into two approaches: deploying new relay nodes to the failure nodes or failure regions to recover the connection and moving some of the original nodes to recover the connection. The method of moving some original nodes to recover the connection does not require the deployment of new nodes, but it does require the nodes to be mobile. For recovery through deploying new nodes, some heuristic methods have been proposed. A previous study proposed a method in which the failure node’s neighboring positions are replaced to recover the network connectivity. However, this method cannot handle cases in which multiple nodes fail at the same time, and some nodes cannot move in some partitions.
In Lloyd and Xue, the minimum spanning tree algorithm based on a single-tiered relay node placement (MST-I-TRNP) uses the Kruskal or the Prim algorithm to find the minimum spanning tree. Then relay nodes are deployed along the edge of the tree based on the length of the relay nodes’ communication radius. However, this method has higher complexity, and the average node degree is low after recovery. To improve the average node connectivity, the simply Connected Spiderweb (1C-SpiderWeb) algorithm is proposed. 1C-SpiderWeb completes the relay node deployment by forming a network structure similar to a spider’s web. Although the average node connectivity increases, this approach requires a much higher number of relay nodes. A cell-based optimized relay node placement (CORP) algorithm divides the network into units of equal size. Then, based on the grid positions in the partition, it deploys relay nodes to recover the connection. The network topology balances the network data transmission capacity and reduces the data transmission delay between partitions. However, the complexity of this approach is high, and it is not suitable for large-scale networks.

System model and formulation of the problem

The WiLD network is expressed as $G = (V, E)$, where $V$ and $E$ refer to the sets of nodes and edges in the network, respectively, and $e_{ij} \in E$ denotes the link between node $v_i \in V$ and $v_j \in V$. Suppose there is a finite number ($N$) of point locations at which to place WiLD nodes in the factory. The point locations capable of housing a WiLD node are denoted as $p_i (i = 1, 2, \ldots, N)$, and $p_i = \begin{cases} \{1\}, & \text{place node} \\ \{0\}, & \text{others} \end{cases}$. Of these locations, $M$ must be assigned nodes that execute the function of gateway nodes; specifically, $p_i = 1, i \in [1, M], M \leq N$ for these $M$ nodes. In this article, assume that the traffic flow $f_k \in F$ of the perceptual and audio/video information collected by each sensor enters the WiLD network via a gateway node and then reaches the control center node $\sigma (\sigma \in V)$ after being transmitted by a WiLD relay node. Feedback information is sent to the relevant gateway node by the control center node $\sigma$ after data analysis and processing and then transmitted to the lower actuator node to execute the specific operation.

Each node in the network is equipped with one or several directive antennas. Here, we assume that the antennas are of the same type and have an identical beam width $\theta$ in this article. To avoid interference between adjacent antennas, a buffer area with an included angle $\Delta_\theta$ is set between adjacent antennas of the same node, and the upper bound on the number of directive antennas that can be configured for each node is $k = \lfloor 360^\circ / (\theta + \Delta_\theta) \rfloor$, as shown in Figure 3. Based on the conclusion in Zenghua et al., the included angle of antennas for the same node should not be less than $30^\circ$. Thus, the process of determining the number and angles of directive antennas of a certain node is the process of determining the connectivity between this node and the surrounding nodes. We define $a_{ij} = \{1, \exists E_{ij} \in E\}, 0$, others, and the number of antennas on each node is $v_i, a_i = \sum_{j\in[1,N]} a_{ij}$, where $0 \leq a_i \leq ((2\pi)/(\theta + (\pi/6)))$. Suppose the basic construction cost of each node is the cost of the node without an antenna, $a$, plus the cost added by each antenna, $b$. Then, the construction cost of each node can be expressed as $c_i = a + b \cdot a_i$. Therefore, the total cost of node construction in the network is $\sum_{i=1}^N c_i$.

Occlusion is measured by the bandwidth of the received signal. Suppose $B_{r}$ is the signal transmission bandwidth of the transmitting terminal and $B_{r}$ is the actual receiving bandwidth of the receiving terminal. Under the same transmission power, the larger the value of $B_{r}$, the smaller the occlusion and the better the channel quality. Conversely, for a smaller value of $B_{r}$, the occlusion will be more serious, and the channel quality will be poorer. When complete occlusion causes connection failure between nodes, $B_{r} = 0$. Suppose that all links are symmetric links. The transmission bandwidth of signal in link $e_{ij} \in E$ is $B_{ij} := B_{ij}^t = B_{ij}^e$. $w_0$ is the end-to-end signal transmission bandwidth of the path that traffic $f_k$ goes through from source node $v_i (i \in [1, M])$ to destination node $\sigma$. The traffic transmission bandwidth on the entire path is the minimum transmission bandwidth of a link on this path, namely, $w_k = \min_{\{\phi_j \in [1, N]\}} w_{0 ij} \cdot x_{ij}$, where $\phi_k \leq w_{0 ij} \leq B_{ij}$ is the transmission rate of traffic $f_k$ on link $e_{ij} \in E$ and $\phi_k$ is the minimum transmission bandwidth demand of traffic $f_k$. $x_{ij} = 1$ indicates that node $v_i$ can be connected to node $v_j$; otherwise, $x_{ij} = 0$. When $x_{ij} = 1$, nodes $v_i$ and $v_j$ can be connected by link $e_{ij} \in E$ directly, or indirectly connected via other nodes.
This article aims to determine the placement locations of WiLD nodes and the number and directions of their antennas in the network to reduce the influence of occlusion on the network and maximize the network throughput while satisfying the traffic delay condition and bandwidth demand and minimize the overhead.

Optimization objective

The optimization objective is to obtain the maximum and minimum values, namely, \( u_1 = \max \sum_{i \in F} W_i \) and \( u_2 = \min \sum_{i=1}^{N} p_i C_i \), respectively. Therefore, we suppose \( u(p, a) = (\sum_{i \in F} W_i)/\left(\sum_{i=1}^{N} p_i C_i\right) \). Then, the optimization objective is max \( u(p, a) \).

Constraint

Real-time performance constraint. According to the analysis of Jintao et al., the delay of each hop is fixed when using the 2P MAC protocol, and we denote it as \( \tau \). Then, the end-to-end delay of traffic \( f_k \) can be expressed as \( \tau \cdot K_{f_k} \), where \( K_{f_k} = \sum_{i=0}^{K_f} W_i \) is the hop count of the path of \( f_k \). \( C_i \) is the link that traffic \( f_k \) passes through. Suppose the end-to-end delay demand of traffic \( f_k \) is \( \mu_k \). Then \( \tau \cdot K_{f_k} \leq \mu_k \).

Bandwidth demand constraint. The bandwidth provided by the transmission path of the traffic must meet the bandwidth demand of the traffic, namely, \( \min_{\forall i,j \in [N]} W_{ij} = x_{ij} \geq d_i \).

Link capacity constraint. The total transmission bandwidth occupied by each traffic on each link should not exceed the bandwidth provided by this link. Thus, \( \forall i,j \in [1,N], j \neq i, \sum_{i=1}^{N} W_{ij} \leq B_{ij} \).

Gateway node setting. For each gateway node \( M \) in the network, \( p_i = 1, i \in [1,M] \).

Node setting constraint. If node \( v_i,j \in [1,N] \) can be connected with node \( v_i,j \in [1,N] \), then node \( v_i \) must be the node that has been set. That is, \( x_{ij} \leq P_i, x_{ij}, P_{ij} \in \{0, 1\} \).

In conclusion, the optimization problem in this article can be considered a special type of the universal capacity limitation facility location (universal facility location) problem. The problem here differs from the universal facility location in that, in this article, traffic overhead is not only relevant to the node from the client side (gateway) to the location under selection, the overhead during the relay between exchange boards and from an exchange board to a data center node should also be considered. It has been proven that the universal facility location problem is a NP-hard problem; consequently, the problem in this article is also an NP-hard problem.

Solutions

Each directional antenna is replaced by an omnidirectional antenna with the same transmission range. Point-to-point communication is adopted between nodes; therefore, the interference between nodes can be ignored. For each node, all nodes that can be connected to it when communicating at maximum power are considered to be nodes connected to this node. Thus, a topological graph of network connections can be constructed. The maximum bandwidth of an information transmission between two nodes connected with an edge is taken as the weight of that edge. According to the analysis of the network operation mode in the above section, the traffic flow bandwidth demand from the gateway node to the data center node is larger than the traffic flow bandwidth demand in the reverse direction. Therefore, if the bandwidth demand from a gateway node to the data center node can be fulfilled in the solution, the total bandwidth demand of the whole network is also satisfied.

The goal of the P-median model is to separately find proper locations for \( P \) facilities and designate for each demand point the given facility under the demand set of the given quantity and location. In addition, the P-median model identifies a set of candidate facility locations, thus obtaining the lowest transport cost between the facility and demand point. The traditional P-median model considers only the single-hop cost between each demand point and facility and is a single-hop single-objective model, whereas in the WiLD network, the end-to-end cost between two demand points should be considered, and it is therefore a multi-hop model. Both bandwidth demand and construction cost objectives are considered in this article; therefore, the proposed model is a multi-objective model. Consequently, to obtain a solution for the WiLD network, a hierarchical solution method based on the P-median model is presented in this article.

Hierarchical traversal solving method based on the P-median model

The total number of nodes in the network is \( N \). Of these \( N \) nodes, \( M \) gateway nodes and a data center node must be deployed. Therefore, the total number of nodes in need of placement is \( N - M - 1 \) at most. The defined variable is \( P \in [1,N - M - 1] \), and the P-median model is established. If the total number of relay nodes cannot be determined, the improved P-median model can be applied to obtain the solution. The hierarchical
traversal solving method based on the P-median model (P-HTM) is applied starting from \( P = N - M - 1 \). The deployment location of each node is traversed under different \( P \) values to determine the maximum network flow under this deployment state until the maximum flow cannot meet the bandwidth demand; \( u(p, a) \) is obtained at this time. Finally, values of \( u(p, a) \) are compared under different \( P \) values to obtain max \( u(p, a) \) of the network.

This algorithm is implemented by adding the restriction condition \( p = \sum_{i=1}^{N-M-1} p_i \) on the uncertain node setting number and adopting a traversal method. The traversal method can obtain the exact solution, but its scouting speed is relatively slow. The P-median model-based relay-node location problem is a discrete problem; therefore, when the number of facility selection points is not large, the computing time of the traversal algorithm is limited.

The calculation of \( u(p, a) \) under each \( P \) value in this algorithm is performed as shown in Algorithm 1. The basic steps of the algorithm are as follows:

---

**Algorithm 1 Available bandwidth calculation**

**Require:** \( G = (V, E), N, M \)

**Ensure:** \( u(p, a) \)

1. for \( \forall \) gateway \( v_i \in V \) do
2. connect \( v_i \) to \( v_j \) with links \( e_{ij} \);
3. \( B_{ij} \leftarrow +\infty \);
4. end for
5. for \( e_j \in E \) do
6. \( w_j \leftarrow 0 \);
7. end for
8. while \( \exists \) augmenting path \( r \) from \( v_i \) to \( \sigma \) do
9. \( r_{df} \leftarrow \) the path starting from \( v_i \) to \( \sigma \) and with minimum hop count;
10. \( BwA(r_{df}, B_{ij}) \);
11. for node from \( \sigma \) to \( s \) do
12. if \( \exists \) bandwidth surplus point \( v_i \) then
13. \( r_{sf} \leftarrow \) the path starting from \( v_j \) to \( \sigma \) and with minimum hop count;
14. \( BwA(r_{sf}, B_{ij}) \);
15. end if
16. end for
17. calculate \( \sum_{k \in f} w_k \) and \( \sum_{i=1}^{N} p_i c_i \);
18. end while

---

**Initialization.** The traffic bandwidth passing through each link is 0 at the beginning. Set a virtual node \( v_i \) in the network and add a link with bandwidth capacity \( +\infty \) between \( v_i \) and each gateway node.

**Construction of augmenting path.** Judge whether there is an augmenting path between \( v_i \) and the control center node \( \sigma \) starting from \( v_j \). If there is such an augmenting path, find a path with the lowest number of intermediate nodes (the shortest augmenting path) from \( v_i \) to \( \sigma \) starting from \( v_j \). Along the link with the most residual bandwidth. When multiple paths have the same intermediate node number, select the augmenting path that can generate the largest augmenting capacity.

**Link bandwidth occupation adjustment.** Set \( \delta \) as the minimum value of residual bandwidth of each link on the selected augmenting path and increase \( \delta \) for bandwidth occupation of all links in the augmenting path.

Conduct reverse query for each node in each path along the selected augmenting path, in order, with \( \sigma \) as the starting node. Suppose that node \( v_i \) on this augmenting path satisfies the following conditions:

1. The input link of \( v_i \) has residual bandwidth, but the output link has no residual bandwidth.
2. There is no such point with residual bandwidth at the input link but without residual bandwidth at the output link before \( v_i \).
3. There is residual bandwidth at the first section of the link on the augmenting path.
4. There is a path from \( v_i \) to \( \sigma \).

Then, \( v_i \) is defined as the surplus point. Keep the path from \( v_i \) to \( v_j \) in the original augmenting path unchanged, select the shortest path from \( v_i \) to \( \sigma \) along \( v_i \), and conduct link bandwidth occupation adjustment for path \( v_i \rightarrow v_j \rightarrow \sigma \) again. The calculation function for link bandwidth is shown in Algorithm 2.

---

**Algorithm 2 Function BwA(r_{df}, B_{ij})**

**Require:** \( G = (V, E), r_{df}, B_{ij} \)

**Ensure:** \( w_j \)

1. for \( e_j \in E \) in path \( r_{df} \) do
2. \( \delta \leftarrow \min \{B_{ij} - w_j\} \);
3. \( w_j \leftarrow w_j + \delta \);
4. end for

---

Regarding the calculation of max \( u(p, a) \), calculate the corresponding \( u(p, a) \) under different \( P \) values and determine the maximum value max \( u(p, a) \).

**Algorithm complexity analysis.** During the execution process for the algorithm, each \( P \) should include searching the augmenting path and adjusting the link bandwidth occupation. When searching the augmenting path via depth-first search, if the last node in the current path can be accessed with link bandwidth extension, the longest length of the augmenting path is \( N \); that is,
node $\sigma$ is reached after advancing $N$ steps to obtain the augmenting path. During reverse query, the number of reverse backoff steps is $N$ at most. Therefore, the number of queries of the augmenting path is $|E|$ at most, and at most $N$ nodes are queried each time. Thus, the total number of query operations is $N + |E| \cdot N$ at most and the complexity is $O(|E| \cdot N)$. The complexity of conducting link bandwidth occupation adjustment in each iteration is $O(N)$. Therefore, when the initial value of $P$ is $N - M - 1$, the overall complexity of the algorithm is $O(|E| \cdot N^2 \cdot (N - M - 1))$.

### 4.2 P-median model-based hierarchical heuristic solving method

To resolve the location problem of nodes when the network size is relatively large, a P-median model-based hierarchical heuristic solving method (P-HHM) is presented. First, let $P = P_{\text{max}} = N - M - 1$ and take the bandwidth that can be provided by the link as the weight to obtain the spanning tree that maximizes the total bandwidth, as shown in Algorithm 3. Then, subtract $(p_{\text{max}} - p)$ nodes in initial spanning tree based on different $P$ values. Connect the residual nodes that are disconnected from the spanning tree to their adjacent nodes in the spanning tree. If there are several such adjacent nodes, select the nodes with the largest connection link bandwidth for connection. Calculate the maximum transmission bandwidth that the network can achieve under this $P$ value until the obtained maximum transmission bandwidth fails to meet the bandwidth demand of traffic.

The calculation process of the P-median-based heuristic solving method is as shown in Algorithm 4, and its main steps are as follows.

**Algorithm 3** Spanning tree function $ST(G, W)$

**Require:** $G = (V, E), g \in V$

**Ensure:** minimum spanning tree edge set $LE$

1: $(d_i^0) \leftarrow \text{sort}(E), v_i, v_j \in V, k \in [1, |E|];$
2: $LE \leftarrow \Phi$
3: $LV \leftarrow \Phi$
4: while $V \neq \Phi$ do
5: if $v_i \not\in LV$ and $v_j \not\in LV$ then
6: $LE \leftarrow LE + \max(d_i^0)$
7: $V \leftarrow V - \{v_i\} - \{v_j\}$
8: $LV \leftarrow LV + \{v_i\} + \{v_j\}$
9: end if
10: end while

**Algorithm 4** P-HHM

**Require:** $G = (V, E), g \in V$

**Ensure:** $\max u(p, a)$

1: $LV \leftarrow ST(G, W)$
2: $\{R_i\} \leftarrow \text{route}(LV)$
3: calculate $u(p, a)$
4: while $P \geq 1$ do
5: if $w \geq \sum_k w_k$ then
6: for $\forall v_i \in R_i$ do
7: $\{R_i\} \leftarrow \{R_i\} - v_i$
8: $\{R_i\} \leftarrow \text{route}(\{R_i\})$
9: calculate $u(p, a)$
10: end for
11: $P \leftarrow P - 1$
12: end if
13: end while

**Complexity analysis of the algorithm.** In the construction process of the spanning tree based on the traditional “kruskal” algorithm, traversing is required for $|E|$ edges. The “While” loop of Algorithm 4 should be executed $N - M - 1$ time(s) at most. In each “While” loop, the nested “for” loop should be executed $P$ time(s), and the maximum value of $P$ is $N - M - 1$. Therefore, the time complexity of the algorithm should not exceed $O((N - M - 1)^2)$.

**Parallel interactive local search algorithm based on dynamic programming**

To resolve the location problem of nodes when the network size is relatively large, we propose a PILS-DP.
Network segmentation

Definition: cut set. Split the \( N - M - 1 \) nodes of the node set \( V \) in the network \( G = (V, E) \) into \( m \) non-empty sets \( V_1, V_2, \ldots, V_m \) such that \( V_1 + V_2 + \cdots + V_m = V \) and \( V_i \cap V_j = \emptyset, \forall i, j \in \{1, 2, \ldots, m\} \). For \( \forall V_i \subseteq V, \) if \( \exists e_{ab} \in E \) and \( v_a \in V_i \) such that \( v_b \in V - V_i \), then we call the network node set \( V_i \) a cut set of \( V \).

Take the minimum hop count \( h_{\text{min}} = \min\{h_{i, \sigma}\}, i \in [1, M] \) from each gateway node \( v_i \) to the data center node \( \sigma \). Divide all nodes except the data center node \( \sigma \) evenly into \( h_{\text{min}} \) cut sets and place all gateway nodes in the same cut set.

Local search in each cut set. For each cut-set \( V_i \), we use the depth-first search method to find any feasible solution that can meet the constraints first and obtain the node settings in this situation. For the nodes in each cut set, we define three operations: add, remove, and node settings in this situation. For the nodes in each cut set, we define three operations: add, remove, and node settings in this situation. For the nodes in each cut set, we define three operations: add, remove, and node settings in this situation. For the nodes in each cut set, we define three operations: add, remove, and node settings in this situation. For the nodes in each cut set, we define three operations: add, remove, and node settings in this situation. For the nodes in each cut set, we define three operations: add, remove, and node settings in this situation.

Dynamic programming inter-cut-set search. The basic idea to solve the problem with dynamic programming is as follows.

The \( h_{\text{min}} \) cut sets obtained from the network segment make up the subproblem of \( h_{\text{min}} \) phases. Each deployment decision of the phase is the subproblem solution in this phase.

<table>
<thead>
<tr>
<th>Algorithm 5 PILS-DP</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Require:</strong> ( G = (V, E), v_i \in V, N, M )</td>
</tr>
<tr>
<td><strong>Ensure:</strong> state variable of each ( V_i )</td>
</tr>
<tr>
<td>1: ( h_{\text{min}} \leftarrow \min{h_{i, \sigma}}, i \in [1, M] )</td>
</tr>
<tr>
<td>2: divide ( N ) into ( h_{\text{min}} ) cut set evenly and all gateways are in one cut set</td>
</tr>
<tr>
<td>3: for ( V ) cut set ( V_i ) do</td>
</tr>
<tr>
<td>4: get a feasible deployment solution ( \hat{V} ) using depth-first search</td>
</tr>
<tr>
<td>5: ( \Pi(\hat{V}) = {\hat{V} + {v_a}, v_a \in V - \hat{V}} \cup {\hat{V} - {v_b}, v_b \in \hat{V}} \cup {\hat{V} + {v_a} - {v_b}, v_a \in V - \hat{V}, v_b \in \hat{V}} )</td>
</tr>
<tr>
<td>6: if ( \exists v_i \in \Pi(\hat{V}), u_{v_i} &gt; u_{v_i} ) then</td>
</tr>
<tr>
<td>7: ( \hat{V} \leftarrow \hat{V} )</td>
</tr>
<tr>
<td>8: end if</td>
</tr>
<tr>
<td>9: record the optimal solution in each situation with edge node as the state variable in each cut set</td>
</tr>
<tr>
<td>10: end for</td>
</tr>
</tbody>
</table>

In each phase, the feasible solutions are obtained by local search in the cut set for the different states (set to \( k \), where \( k = 1, 2, 3, \ldots \)). The state variable is denoted by \( s_k \), and the value of is the \( k \) possible alternative feasible solution.

In each state \( s_k \), the decision-making variable is \( \xi_k(s_k) \). The values of \( \xi_k(s_k) \) are the elements in the state variable set.

The index function of the cost measured in the \( h_i(i = 1, 2, \ldots, h_{\text{min}}) \) phase is \( v_{k,n} \). \( v_{k,n}(s_k, x_k) \) is the accumulation function from state \( s_k \) to the decision-making variable \( x(s_k) \) and all subsequent stages.

The optimal value of the index function is the optimal index function, denoted by \( f_k(s_k) \), and \( f_k(s_k) = \text{OPT} \{v_{k+1}(s_k, \xi_k), f_{k+1}(s_k, x_k + 1)\} \). This is the basis for decision-making under the state \( s_k \).

According to the overall optimal solution and the reverse solving ideas of dynamic programming, we get the decision \( \xi_k(s_k) \) of each phase, and we can then obtain the ordered overall optimal decision \( p_{1, h_{\text{min}}} \), that is, \( p_{1, h_{\text{min}}} = \{\xi_1(s_1), \xi_2(s_2), \ldots, \xi_n(s_{h_{\text{min}}})\} \).

The complexity of the algorithm. A depth-first search is carried out in each cut set. In the worst case, it is not until the last node that the feasible solution is obtained when using the depth-first search method to search for the feasible solution satisfying the constraints. Thus, the total computational complexity of the search is \( O((N - M - 1)/h_{\text{min}})^2) \).

The time complexity of the dynamic programming algorithm is \( O(\sum_{i=1, h_{\text{min}}} k_i h_i)^2 \). Because \( \sum_{i=1, h_{\text{min}}} k_i h_i \leq h_{\text{min}} \sum_{i=1, h_{\text{min}}} k_i \leq h_{\text{min}}(N - M - 1) \), and in the worst cases, \( h_{\text{min}} = 1 \), the overall complexity of PILS-DP is \( O((N - M - 1)^2) \).
Network node deployment for traffic bursts

When the original network layout cannot meet the transmission demands due to traffic bursts, we can recover the network’s performance by increasing the number of nodes or the number of antennas in some nodes to increase the network bandwidth and meet the traffic transmission demands. Thus, we propose a network BRM-ST. The basic idea of this method is that when the bandwidth is insufficient, we use the residual bandwidth that another path can provide to transmit; if we cannot find another path with sufficient residual bandwidth to meet the demands, we consider the link that cannot meet the bandwidth demands to be disconnected, and the network is partitioned. Then, we structure the network partition connectivity based on the Steiner tree28 to meet the bandwidth transmission demands.

One possible approach is to select the node with the minimal cost path to the Steiner tree and add the node to the tree through the minimal cost path. However, the time complexity of iterating to find the possible paths is too high. To reduce the time complexity, we use the basic idea of the progressive increase in the Dijkstra algorithm to define the node search order and skip the feasible paths with higher cost to achieve a fast search. The algorithm mainly includes the following three stages: preparation, Steiner tree construction, and fast searching.

Preparation

The locations where nodes can be placed are assumed to have nodes equipped with omni-directional antennas and maximum power. When the transmission bandwidth of a link is less than the demand of the traffic burst, the link is seen as disconnected, and the network is partitioned. All partitions segi form the set $D$, ordered from large to small order according to distance (Dist(segi, $a$)) from the data center node $a$. All the locations that can accept a node between each partition form the set $Q$, and the number of such locations is $n$.

Steiner tree construction

Define the path weight of the path $P_i$ as $g_i = \lambda / w_i$, where $w_i$ is the path bandwidth, whose value is the minimum of all the link bandwidths in this path; that is, $w_i = \min_{e \in P_i} \{w_e\}$, and $\lambda$ is a constant.

Using the Dijkstra shortest path tree (Dijkstra-SPT) algorithm,10 we can get the minimum path weighted Steiner tree $T$ rooted in $a$ for the current network connection.

Fast searching

In the fast-searching stage, we introduce two variables $\delta$ and $\pi$ for each node in the network. $\delta(a)$ records the minimum-cost path from node $va$ to $T$, and $\pi(a, \delta)$ records the parent node of $va$ in the minimum-cost path tree. The search starts from the data center node $\delta$ and proceeds to gateway nodes that have traffic burst demands.

The minimum cost from node $va$ to the Steiner tree is defined as follows: for a given Steiner tree $T$ rooted in node $\delta$, the cost of node $va$ reaching $T$ through the path $P(a, T)$ is the cost without the shared link cost, that is, $\text{cost}(a, P) = \sum_{e \in P, e \in T} \text{cost}(e)$. Its value represents the added value to $T$ when connecting to $T$ through the path $P$.

The path weight set from $va$ to $T$ for weights less than or equal to $\lambda / d_{va}$ is denoted as $S_p(a, T) = \{P | \text{length}(P) \leq (\lambda / d_{va})\}$. In addition, the path cost from $va$ to $T$ less than or equal to $\lambda / d_{va}$ is $\delta(a) = \min \{\text{cost}(a, P) | P \in S_p(a, T)\}$. Thus, we can get the minimum feasible cost from $va$ to $T$, that is $\delta(a, T) = \min(\delta(a))$, and the corresponding path is the minimal cost path from $va$ to $T$.

The recursion relation of $\delta(a)$ is

\[
\delta(va) = \begin{cases} 0, & va = a, g_a = 0 \\ \delta(va) + \min \{\delta(w, g_a) + \text{cost}(w, va) \mid w \in \text{adjacency}(va) \text{ and } va \neq s, g_i > 0\}, & \text{others} \end{cases}
\]

In the fast-searching stage, we search the node with minimum $\delta$ (which has the minimum-cost path to the current tree) in turn. If the node is the gateway node, the node joins the path tree through the minimum-cost path. The main idea of the fast-searching stage is as follows.

Initialization. The current time delay constraint, Steiner tree $T = \emptyset$, $\delta(a, 0) = 0$, $\delta(a, g_a) = \infty$, $va \neq a$, $\pi(a, a) = 1$ (line 1 in Algorithm 6).

Choose the node with the minimum cost to $T$ from $Q$. If multiple nodes are optimal, choose the node with maximum path bandwidth; denote it as $va$. The minimum feasible path weight is $g_a$ (line 2 in Algorithm 6).

If $va$ is the gateway node, let $vb = va$, $k = g_a$. If $vb \not\in T$, and the corresponding edge $e(b, \pi(b, k))$ will join $T$, and let $\delta(vb, k) = 0$. If $vb$ has been searched, then $vb$ will rejoin $Q$, along with $\pi(vb, k) = k = g(c, b)$, $vb = vc$ and repeat until $vb \in T$ (lines 3-16 in Algorithm 6).

For each neighbor node $vb$ of $va$ (denoted as $vd \in \text{adjacent}(va)$), let $l = g_i + g(a, d)$. If $vd \not\in T$, $\delta(d, l) > \delta(a, g_i) + \text{cost}(a, d)$, update the cost information from $vd$ to $T$, and let $\delta(vd, l) = \delta(va, g_i) + \text{cost}(va, vd)$, $\pi(d, l) = va$. If $vd$ has been searched, $\delta(d, l) < \delta(a, g_i) + \text{cost}(a, d)$.
Algorithm 6 Fast searching

Require: Q, \( |g| \geq 0 \)
Ensure: T
1: \( T \leftarrow \emptyset, \delta(v_0) \leftarrow \infty, \pi(v_0, g) \leftarrow -1; \)
2: choose \( v_0 \in Q \) with the min cost \( (v_0, T) \);
3: if \( v_0 \) is the gateway node with growing bandwidth demand then
4: \( v_0 \leftarrow v_0, k \leftarrow g; \)
5: while \( v_0 \notin T \) do
6: \( T \leftarrow T + \{v_0\}; \)
7: \( \delta(v_0, k) \leftarrow 0; \)
8: if \( v_0 \) has been searched then
9: \( Q \leftarrow Q + \{v_0\}; \)
10: \( v_0 \leftarrow \pi(v_0, k), k \leftarrow k - g(v_0, v_0), v_0 \leftarrow v_0; \)
11: end if
12: end while
13: end if
14: if \( v_0 = \sigma \) then
15: \( T \leftarrow T + \{v_0\}; \)
16: end if
17: for \( \forall v_d \in \text{adjacent}(v_0) \) do
18: \( l \leftarrow g + g(v_0, v_d); \)
19: if \( v_d \notin T \& \& \delta(v_d, l) > \delta(v_0, g) + \text{cost}(v_0, v_d) \) then
20: \( \delta(v_d, l) \leftarrow \delta(v_0, g) + \text{cost}(v_0, v_d); \)
21: \( \pi(v_d, k) \leftarrow v_0; \)
22: end if
23: if \( \delta(v_d, l) \) is the minimum deployment cost from \( v_d \) to \( T \& l \) is the minimum feasible path weight then
24: \( Q \leftarrow Q + \{v_0\}; \)
25: end if
26: end for
27: \( Q \leftarrow Q - \{v_0\}; \)

is the minimum deployment cost and \( l \) is the minimum feasible path weight, then \( v_d \) will rejoin \( Q \) (lines 17–26 in Algorithm 6).

Then, delete \( v_d \) from the set \( Q \) (line 27 in Algorithm 6). If multiple nodes have traffic burst demands, repeat the above steps until \( T \) covers all these nodes and the algorithm ends.

Finally, check the results and delete the duplicate nodes and antennas that can communicate in the feasible bandwidth to further reduce the number of node deployments.

The complexity of the algorithm

In the Steiner tree construction stage, we use the Dijkstra-SPT algorithm to establish the spanning tree. Thus, the complexity is \( o(n^2) \). In the fast-searching stage, we choose the most appropriate nodes from the node set for the next extension, which is to find the node with the minimum distance from the vector. The complexity depends on the data structure of the selected algorithms. In this article, we use an array to implement it; therefore, the time complexity is \( o(n) \). In the process of adding the node through the minimal cost path to Steiner tree \( T \), the time complexity is \( o(l_a) \), where \( l_a \) is the number of nodes in the path. In the process of the search loops, the total number of loops is the total number of nodes to search. Although there are some nodes that are searched again, previous experiments showed that the average search frequency of a node is less than 2, so the time complexity of the loops is \( o(n) \). Overall, the time complexity of BRM-ST is \( o(n^2) \), where \( n \) is the number of remaining locations where the nodes can be deployed.

Experiment and analysis

This section describes a simulation experiment conducted to verify the performance of the algorithm proposed in this article. We used MATLAB to simulate the performance of our algorithms. A network is randomly generated in a 2 km \( \times \) 2 km area, points in the network are randomly selected for the data center node and the gateway node, and traffic is sent to the data center node via the gateway node. The maximum transmission bandwidth and minimum network construction overhead of the network under different conditions are obtained by changing the maximum number of selectable point locations, the transmission power of the nodes, and the traffic bandwidth demand in the network. The maximum transmission bandwidth of traffic between nodes is set to 30 M. The bandwidth between nodes decreases linearly as the distance between the nodes increases. Because there is no similar algorithm that considers the same two objectives (network throughput and construction cost), we compare our algorithms’ results with the existing GA. We simulate the computation times, the maximum bandwidth provided under different numbers of selectable point locations, and the number of nodes required under different traffic bandwidth transmission demands and the maximum transmission bandwidth with these algorithms. In addition, a comparison is made between the P-HTM and P-HHM algorithms presented in this article and GA. Because as we know that this is the first article that considers two objects of nodes layout with IEEE 802.11 long-distance 2P MAC, there are no techniques available in the literature, and we just compare our methods with the traditional GA. The test results are shown in Figures 4 and 5.

Figure 4 shows the computing time of each algorithm for solving the problem based on different numbers of selectable point locations. It can be observed from the figure that the computing times of the P-HHM and PILS-DP algorithms are lower than that of GA and that of the P-HTM is higher than GA. In addition, as the number of selectable point locations increases, the computing times of the P-HTM algorithm and GA algorithm increase rapidly, but that of the P-HHM and PILS-DP algorithms increase slowly.
Figure 5(a) shows the maximum bandwidth that the network can provide under different numbers of selectable point locations. Figure 5(b) shows the number of nodes required for network transmission under different traffic bandwidth transmission demands. Figure 5(c) shows the total number of nodes required by network with different maximum transmission bandwidths that can be provided by each node when the transmission bandwidth demand of network traffic is 120. It can be seen from Figure 5 that the results from the P-HTM, P-HHM, PILS-DP, and GAs are basically identical. Therefore, the simulation result shows that an exact solution can be obtained via P-HHM and PILS-DP algorithms, which also take less computing time. While with the parallel computing feature, PILS-DP can further reduce the computing time.

To further verify the performance of the algorithm, a network is set up in Fushun Olefin Factory, Sinopec Group for a test, as shown in Figure 6. Plants and equipment in the factory are distributed within the scope of 1.5 km $\times$ 1.5 km, and occlusions include buildings, metal tanks, and pipelines in the factory. There are 14 points in the factory that can be selected...
to set up nodes, and two of the nodes should be connected to gateway nodes.

A test is conducted under different bandwidth demands of two gateway nodes, and the required number of nodes is obtained by applying different algorithms, as shown in Figure 7. Figure 7 shows that the solution of the network model can be obtained via both the P-HTM and P-HHM algorithms.

In transmission performance recovery experiments, we compare our proposed algorithm based on a Steiner tree with the traditional minimum spanning tree–based algorithm in terms of the number of relay nodes deployed and the average node degree when meeting the transmission demands, as shown in Figures 8–10.

Figure 8 describes the impact on the number of relay nodes’ deployment of the partition number. We can see from the figure that with the increase in the number of partitions, the required number of relay nodes of the two algorithms increases. This trend corresponds to the actual situation. An increase in the number of partitions will lead to the use of more relay nodes when connectivity is recovered. However, we can also see clearly from the figure that the proposed BRM-ST
algorithm requires fewer relay nodes than the traditional minimum spanning tree–based algorithm, and with an increase in the number of partitions, this advantage will be more obvious. This phenomenon occurs because with an increase in the number of partitions, the Steiner tree can connect a much higher number of partitions, which will reduce the number of relay nodes required.

Figure 9 describes the impact of the communication radius on the number of relay nodes required when the partition number is 9. We can see from the figure that as the communication radius increases, the number of relay nodes required is reduced. This trend also corresponds with the actual situation because with the increase in the communication radius, the number of relay nodes required will decrease when the path length is fixed. At the same time, we can see from the figure that the number of relay nodes required by the BRM-ST algorithm proposed in this article is less than that required by the algorithm based on the minimum spanning tree. Of course, this advantage is not obvious because the increasing communication radius will lead to a decrease in the path length.

Figure 10 describes the impact of the size of the partition on the number of relay nodes required when the partition number is 9 and the communication radius is 20 km. We can see from the figure that with an increase in partition size, the number of relay nodes required increases. This result occurs because the increase in partition size will lead to an increase in the distance between partitions, thus the number of relay nodes required will increase. We can also see clearly from the figure that the number of relay nodes required by the proposed algorithm BRM-ST is less than that by the algorithm based on the minimum spanning tree, and with the increase in the partition size, these advantages are more obvious. This result is because the sum of the distances required by BRM-ST is smaller than that by the minimum spanning tree, and with the increase in the partition size, this advantage will be more obvious.

Conclusion

The goal of this article is to solve the layout problem of the P-median model for a heterogeneous industrial monitoring network based on a WiLD multi-hop network architecture. The traditional P-median model cannot be applied to optimize a multi-hop network or a multi-objective network; thus, a hierarchical P-median optimization method based on the traditional P-median model was proposed in this article. The network node layout was optimized by taking the network throughput and construction cost as optimization objectives and using network delay as a constraint. To resolve the node location problem when the network size is relatively large, a PILS-DP is proposed. Then, when the original layout of the network cannot meet the transmission demands due to traffic bursts, we recover the network’s performance using a network BRM-ST. A simulation experiment was performed using a network constructed in a practical environment. Furthermore, we compared the solution obtained by the algorithms proposed in this article with that obtained by the traditional GA-based heuristic method, thus verifying the performance of the proposed algorithms.

Acknowledgements

The authors thank the 12th International Conference on Wireless Algorithms, Systems, and Applications (WASA 2017) for providing an opportunity to preliminarily present the deployment problem and its preliminary solution. The authors also thank the anonymous reviewers for their constructive advice.

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 was supported by the National Natural Science Foundation of China (grant nos 61502474, 61501447, and 61233007) and Youth Innovation Promotion Association of the Chinese Academy of Sciences.

References

procedure of process industrial. *Inform Control* 2016; 45(1): 40–44.


