Closed Walk Ferry Route Design for Wireless Sensor Networks

Qiang Dou[1], Yong Wang[1], Wei Peng[1] and Zhenghu Gong[1]
[1]College of Computer, National University of Defense Technology,
Changsha, Hunan, China
[e-mail: {douqiang, wangyong, wpeng, gzh}@nudt.edu.cn]
*Corresponding author: Yong Wang

Received March 6, 2013; revised September 2, 2013; accepted September 21, 2013; published October 29, 2013

Abstract

Message ferry is a controllable mobile node with large capacity and rechargeable energy to collect information from the sensors to the sink in wireless sensor networks. In the existing works, route of the message ferry is often designed from the solutions of the Traveling Salesman Problem (TSP) and its variants. In such solutions, the ferry route is often a simple cycle, which starts from the sink, access all the sensors exactly once and moves back to the sink. In this paper, we consider a different case, where the ferry route is a closed walk that contains more than one simple cycle. This problem is defined as the *Closed Walk Ferry Route Design* (*CWFRD*) problem in this paper, which is an optimization problem aiming to minimize the average weighted delay. The *CWFRD* problem is proved to be NP-hard, and the *Integer Linear Programming* (*ILP*) formulation is given. Furthermore, a heuristic scheme, namely the *Initialization-Split-Optimization* (*ISO*) scheme is proposed to construct closed walk routes for the ferry. The experimental results show that the *ISO* algorithm proposed in this paper can effectively reduce the average weighted delay compared to the existing simple cycle based scheme.

Keywords: message ferry; closed walk; route design; integer linear programming; wireless sensor networks (WSNs)

http://dx.doi.org/10.3837/tiis.2013.10.002

1. Introduction

Wireless Sensor Networks (WSNs) are wildly used in critical areas, such as environmental monitoring [1], habitat monitoring [2,3], battlefield surveillance, and traffic monitoring [4]. Typically, the data sensed by the sensor nodes needs to be transferred to a base station (sink), where the data could be further analyzed. The traditional approach for data collection in WSNs involves multi-hop communication from sensors to the sink. Under such a scheme, the nodes near the sink have to relay the data from nodes that are farther away. This leads to a non-uniform depletion of energy and the nodes near the sink will run out of batteries very soon due to the limited energy of the sensors. When these nodes die, the network is broken down since the data cannot be transmitted to the sink anymore. In addition, multi-hop communication may not always be possible due to network partitioning in sparse WSNs. Various approaches have been proposed in recent years to overcome the drawbacks of multi-hop communication in WSNs mentioned above. One of them, namely the Message Ferry (or ferry for short) scheme [5,6], uses a controllable mobile node, equipped with large capacity and rechargeable battery, as a ferry to collect the data from the sensors and deliver them to the sink.

In the message ferry scheme [5-9], the ferries move according to a pre-defined route to collect and distribute data from one node to another in a partitioned network. There are some other similar definitions, such as data MULEs [10], mobile elements [1,11], mobile agents [12], mobile sink [13]. [14] considered the message ferry route design for sparse ad hoc networks where each node randomly moves in a certain area. As the node is moving, the solution did not guarantee that the ferry meets the node in each tour, but the contact occurs with certain probability in each tour. In [10], data generated by sparsely located sensors are buffered at the sensors. The MULEs equipped with transceivers act as mobile elements and move randomly to collect data from the sensors when approaching them. The collected data are then carried to a wireless access point. The movements of MULEs are not controlled in this framework. [15] investigated the Mobile Element Scheduling (MES) problem. In that work, according to its deadline, a node needed to be visited multiple times before all other nodes are visited. And after the node is visited, its deadline for its next visit is updated to avoid buffer overflow. A Partition Based Scheduling (PBS) algorithm was proposed for the MES problem in [11]. This algorithm tackled the problem by dividing it into two sub-problems: Partitioning and Scheduling. The nodes were first partitioned into sub-sets according to their data generation rate and location. Then each subset is scheduled individually. The entire route is formed by the route of each subset. The Message Ferry Route (MFR) problem to minimize the average weighted delay is formulated by Zhao et al. in [5]. Ting and Chor investigated the general message ferry route (MFR*) problem in [16] aiming to minimize the average weighted delay in ad hoc networks. They proved that the MFR* problem is MAX-SNP-hard, and proposed the An-Improved-Route (AIR) scheme to solve this problem.

Researchers have attempted to solve the MFR problem and its extentions for different scenarios by adopting solutions from the well-studied Traveling Salesman Problem (TSP) [17,18]. In most of these solutions, the ferry route is designed as a simple cycle, where the ferry starts from the sink, access all the sensors exactly once and moves back to the sink. However, to the best of our knowledge no studies have been carried out to ascertain whether using only one simple cycle as the ferry route will incur less average message delay than closed walks (which may contain more than one simple cycle). The simple cycle route is illustrated as *Route 1* in **Fig. 1**. *Route 1* starts from the sink *O* , consecutively accesses all the sensors *A*, *B*, *C*, *D*, *E* and *F*, and moves back to the sink *O*. There is no accessing of *O* except the start and end in the route. In contrast, a closed walk ferry route could be designed as *Route 2* for the same scenario in Fig. 1. In *Route 2*, the ferry will access the sink *O* after accessing parts of the nodes (i.e., *A*, *B* and *C*), then continues to access the remaining nodes (i.e., *D*, *E* and *F*). Compared to *Route 1*, in *Route 2*, the delivery time (to the sink) of the messages from sensors *D*, *E* and *F* is delayed since $CO + OD > CD$, but the messages from sensors *A*, *B* and *C* are delivered ahead of time. If the total reduced weights of the delivery time of the messages from *A*, *B* and *C* are greater than the increased weights of the delivery time of the messages from *D*, *E* and *F*, the total average weighted delay will be reduced. From this illustration we can infer that, a ferry route containing more than one simple cycle may reduce the delay compared to a simple cycle route. Then the ferry route design scheme when the route could contain more than one simple cycle is to be discussed in this paper, which is defined as the *Closed Walk Ferry Route Design* (*CWFRD*) problem. The aim of the *CWFRD* problem is to minimize the average weighted delay of the messages.

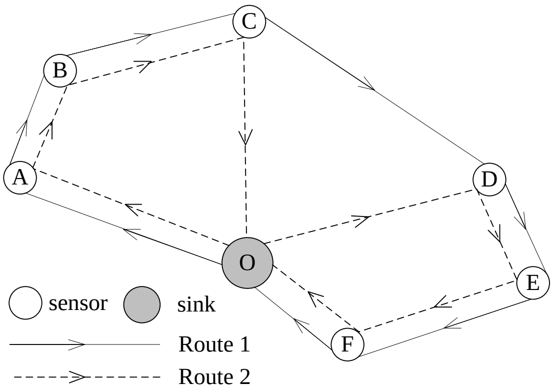

Fig. 1. Ferry Routes Illustration

The main contributions of this work are summarized as follows:

- The *CWFRD* problem is defined in this paper.
- We prove that the *CWFRD* problem is NP-hard, by reducing the classical NP-hard problem TSP to it.
- The *Integer Linear Programming* (*ILP*) formulation of the *CWFRD* problem is given, so that this problem can be efficiently and optimally solved using the existing *ILP* techniques (or tools) for small instances.
- An efficient heuristic algorithm *Initialization-Split-Optimization* (*ISO*) is proposed to solve the *CWFRD* problem by constructing closed walk routes for the ferry to reduce the average weighted delay.

The rest of this paper is organized as follows: The *CWFRD* problem is proposed in section 2. In section 3, the NP-hardness of the *CWFRD* problem is proved. Section 4 gives the *ILP* formulation of the *CWFRD* problem. The heuristic algorithm *ISO* is proposed in section 5. Section 6 presents the experimental results. Finally, the conclusion of this paper is made in section 7.

2. The CWFRD Problem

2.1 Preliminaries

The walk and simple cycle in Graph Theory are defined in [19] as:

Definition 1 (***walk***): A walk is an alternating sequence of vertices and edges, beginning and ending with a vertex, where each vertex is incident to both the edge that precedes it and the edge that follows it in the sequence, and where the vertices that precede and follow an edge are the end vertices of that edge. A walk is closed if its first and last vertices are the same, and open if they are different.

Definition 2 (***simple cycle***): A simple cycle is a walk that starts and ends at the same vertex but otherwise has no repeated vertices or edges.

In this paper, we use $\mathbb{R}$ to denote a closed walk route (or route for short) and use R to denote the simple cycle in $\mathbb{R}$ (or sub-route). R_m represents the m^{th} sub-route of $\mathbb{R}$.

Usually, a closed walk is like the form $[O \to A \to B \to A \to C \to O]$. But in this study, the route of the ferry is a special closed walk which contains no less than one simple cycle. It means that the repeated nodes is also the start vertex (sink). For example, in Fig. 1, the closed walk route $\mathbb{R}_c = [O \to A \to B \to C \to O \to D \to E \to F \to O]$ can be deemed as the combination of two simple cycle routes $R_1 = [O \to A \to B \to C \to O]$ and $R_2 = [O \to D \to E \to F \to O]$.

In the ferry route optimization problems, different applications are interested in different objectives, such as ferry route length, message delay. In this paper, we concentrate to the average weighted message delay. The weighted delay in a network is usually defined as $D = \sum_{i=1}^{N} w_i d_i$, where w_i is the weight (data size) of the message m_i, and d_i is the delay time of m_i. Then the average weighted delay which we are concerned with is $\overline{D} = \frac{D}{\sum_{i=1}^{N} w_i} = \frac{\sum_{i=1}^{N} w_i d_i}{\sum_{i=1}^{N} w_i}$. And let t_i^g and t_i^d denote the generation time and delivery time of m_i respectively. Then the average weighted message delay can be represented as

$$\overline{D} = \frac{\sum_{i=1}^{N} w_i (t_i^d - t_i^g)}{\sum_{i=1}^{N} w_i}. \tag{1}$$

2.2 Assumption

In this paper, we consider the scenario described as follows: All the data sensed by the sensors need to be sent to the sink. The sensors are randomly deployed in a large area. The topology of the network could be flat or hierarchical. When the topology is hierarchical, the ferry only needs to access the cluster heads, and the data generated by the other sensors in the cluster are all transmitted to it through multi-hop. Then the weights (data sizes) of the messages in the sensors are different. There is only one ferry for data collecting in the system, and the ferry is equipped with sufficient buffer and rechargeable energy. The data generation cycles of the sensors are all the same, and this cycle is greater than the time that the ferry accesses all the sensors. This assumption simplifies the problem, and we only need to consider a snapshot of the time evolving data generation. When the data are ready, the ferry starts from the sink. At time 0, the ferry is located at the sink. Then the generation time of the messages are all the same, i.e., $t_i^g = 0$. Then the average weighted delay can be represented as

$$\overline{D} = \frac{\sum_{i=1}^{N} w_i t_i^d}{\sum_{i=1}^{N} w_i}. \tag{2}$$

In addition, the following assumptions are made regarding the ferry and the sensors.

- The physical sizes and the transmission ranges of the sensors and the ferry are negligible, and the ferry needs to move to the location of sensors to begin data collection.
- The ferry can move in any direction without any delay by making any turns.
- Data transfer time between sensors and ferry is negligible compared to the travelling time of the ferry, and the ferry does not need to hover above the sensors to wait for the finish of data transmission.
- The speed of the ferry is a constant value.

2.3 Problem Statement

Let $S = \{s_1, s_2, \cdots, s_N\}$ denote the set of the sensors, where N is the total number of the sensors. And we use s_0 to denote the sink. Let w_i denote the data size of s_i. Then the *CWFRD* problem could be stated as follows:

Problem 1 (***CWFRD***): Given the set of sensors S, the sink s_0, the data size w_i, the aim is to determine a closed walk ferry route, while minimizing the average weighted delay $\overline{D}$, i.e.,

$$\min \frac{\sum_{i=1}^{N} w_i t_i^d}{\sum_{i=1}^{N} w_i} \tag{3}$$

3. Proof of NP-Hardness

In this section, we shall prove that the *CWFRD* problem is NP-hard. The proof relies on a reduction from the Travelling Salesman Problem.

Theorem 1 : The *CWFRD* problem is NP-hard.

Proof: Let $L(s_i, s_j)$ denote the Euclidian Distance between s_i and s_j, i.e., $L(s_i, s_j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$, where (x_i, y_i) is the coordinate of s_i in the two dimensional space. Let v denote the speed of the ferry.

Consider the following scenario: Let $L_{max} = \max_{\substack{\forall s_i, s_j \in S \\ s_i \neq s_j}} L(s_i, s_j)$. Deploy the sensors far away from the sink, satisfying that $\min_{\forall s_i \in S} L(s_0, s_i) \geq \frac{(N^2+4N)L_{max}}{8}$. The data sizes in all the sensors are all the same. Without loss of generality, let $w_i = 1$, then

$$\overline{D} = \frac{\sum_{i=1}^{N} t_i^d}{N}. \tag{4}$$

Now we will show that under this scenario, the minimum average weighted delay exists when there is only one sub-route.

Let $\mathbb{R}^M$ denote the optimal route when when the route contains M sub-routes. Then $\mathbb{R}^M = [\mathbf{s_0} \to s_1^1 \to s_2^1 \to \cdots \to s_{N_1}^1 \to \mathbf{s_0} \to s_1^2 \to s_2^2 \to \cdots \to s_{N_2}^2 \to \mathbf{s_0} \to \cdots\cdots \to \mathbf{s_0} \to s_1^M \to s_2^M \to \cdots \to s_{N_M}^M \to \mathbf{s_0}]$, where s_a^b denotes the a^{th} node of the b^{th} sub-route. And we denote the average weighted delay of route $\mathbb{R}^M$ as $\overline{D}^M$, which is the minimum average weighted delay when the sensors are divided into M subsets. Now we remove the last second s_0 from $\mathbb{R}^M$ and obtain the new route $\mathbb{R}'^{M-1} = [\mathbf{s_0} \to s_1^1 \to s_2^1 \to \cdots \to s_{N_1}^1 \to s_1^2 \to s_2^2 \to \cdots \to s_{N_2}^2 \to s_1^3 \cdots\cdots \to s_{N_{M-1}}^{M-1} \to s_1^M \to$

$s_2^M \rightarrow \cdots \rightarrow s_{N_M}^M \rightarrow \mathbf{s_0}]$. And we denote the average weighted delay of route R'^{M-1} as $\overline{D'}^{M-1}$.

Let $\mathbb{R}^{M-1}$ denote the optimal route when the route contains $M-1$ sub-routes. And we denote the average weighted delay of route $\mathbb{R}^{M-1}$ as $\overline{D}^{M-1}$, which is the minimum average weighted delay when the sensors are divided into $M-1$ subsets.

Now we will compare $\overline{D}^M$ and $\overline{D}^{M-1}$. Since $\overline{D} = \frac{\sum_{i=1}^{N} t_i^d}{N}$ and the delivery time of the messages in the same sub-route are the same, then

$$\overline{D} = \frac{\sum_{m=1}^{M} N_m T_m}{N}, \tag{5}$$

where T_m is the time that the ferry finishes the m^{th} sub-route R_m and N_m is the number of nodes in R_m. Now to compare $\overline{D}$, we could only need to consider the time that the ferry reaches the sink of each sub-route.

The nodes in the first $M-2$ sub-routes of $\mathbb{R}^M$ and $\mathbb{R}'^{M-1}$ are the same, and the delivery times of these nodes are the same. Assume that the delivery time of the nodes in sub-route R_{M-1} of $\mathbb{R}^M$ is T_{M-1}, and then the delivery time of the messages of the nodes in sub-route R_M of $\mathbb{R}^M$ is $T_M = T_{M-1} + \frac{L(R_M)}{v}$, where $L(R_M)$ is the length of R_M. But in $\mathbb{R}'^{M-1}$, the delivery time of the messages of the nodes in sub-route R_{M-1} of $\mathbb{R}^M$ will be increased, since the ferry needs to access the remaining nodes before reaching the sink. The new delivery time of the messages of these nodes is $T'_M = T'_{M-1} + \frac{L(R_M) - L(s_{N_{M-1}}^{M-1}, s_0) - L(s_0, s_1^M) + L(s_{N_{M-1}}^{M-1}, s_1^M)}{v}$.

Then the increased weighted delivery time of the messages in sub-route R_{M-1} of $\mathbb{R}^M$ is

$$\begin{aligned}\Delta_{inc} &= N_{M-1}(t'^M_d - t^{M-1}_d) \\ &= \frac{N_{M-1}\left(L(s_1^M, s_2^M) + \cdots + L(s_{N_M}^M, s_0) + L(s_{N_{M-1}}^{M-1}, s_1^M) - L(s_{N_{M-1}}^{M-1}, s_0)\right)}{v}.\end{aligned}$$

According to the triangle inequality, $L(s_{N_M}^M, s_0) - L(s_{N_{M-1}}^{M-1}, s_0) \le L(s_{N_{M-1}}^{M-1}, s_{N_M}^M)$, then

$$\begin{aligned}\Delta_{inc} &\le \frac{N_{M-1}\left(L(s_1^M, s_2^M) + \cdots + L(s_{N_{M-1}}^M, s_{N_M}^M) + L(s_{N_{M-1}}^{M-1}, s_1^M) + L(s_{N_{M-1}}^{M-1}, s_{N_M}^M)\right)}{v} \\ &\le \frac{N_{M-1}(N_M + 1)L_{max}}{v}.\end{aligned}$$

And the decreased weighted delivery time of the messages in sub-route R_M of $\mathbb{R}^M$ is

$$\begin{aligned}\Delta_{dec} &= N_M(t^M_d - t'^M_d) \\ &= \frac{N_M\left(L(s_{N_{M-1}}^{M-1}, s_0) + L(s_0, s_1^M) - L(s_{N_{M-1}}^{M-1}, s_1^M)\right)}{v}. \\ &\ge \frac{N_M\left(L(s_{N_{M-1}}^{M-1}, s_0) + L(s_0, s_1^M)\right) - N_M L_{max}}{v}\end{aligned}$$

Then the changed average weighted delay is

$$\begin{aligned}\overline{D}^M - \overline{D'}^{M-1} &= \frac{\Delta_{dec} - \Delta_{inc}}{N} \\ &\ge \frac{N_M\left(L(s_{N_{M-1}}^{M-1}, s_0) + L(s_0, s_1^M)\right) - N_M L_{max} - N_{M-1}(N_M+1)L_{max}}{Nv}.\end{aligned}$$

Since $N_M + N_{M-1} \le N$, then $N_M + N_{M-1}(N_M + 1) \le \frac{N^2+4N}{4}$. Then

$$\overline{D}^M - \overline{D'}^{M-1} \geq \frac{N_M\left(L(s^{M-1}_{N_{M-1}}, s_0) + L(s_0, s^M_1)\right) - \frac{(N^2+4N)L_{max}}{4}}{Nv}.$$

Because $N_M \geq 1$,

$$\overline{D}^M - \overline{D'}^{M-1} \geq \frac{L(s^{M-1}_{N_{M-1}}, s_0) + L(s_0, s^M_1) - \frac{(N^2+4N)L_{max}}{4}}{Nv}.$$

As $L(s^{M-1}_{N_{M-1}}, s_0) \geq \frac{(N^2+4N)L_{max}}{8}$ and $L(s_0, s^M_1) \geq \frac{(N^2+4N)L_{max}}{8}$, then we have $\overline{D}^M \geq \overline{D'}^{M-1}$. Since $\overline{D}^{M-1}$ is the minimum delay when there are $M-1$ subsets, then $\overline{D}^{M-1} \leq \overline{D'}^{M-1}$. So $\overline{D}^M \geq \overline{D}^{M-1}$, i.e., under the scenario given above, the less sub-routes $\mathbb{R}$ contains, the smaller average weighted delay is. From this conclusion, we conclude that under the scenario given above, the minimum average weighted delay exists when $\mathbb{R}$ contains only one sub-route.

We use $\mathbb{R}^1$ to denote optimal closed walk route when all the nodes are accessed in only one sub-route, and let L^1 to denote the length of $\mathbb{R}^1$. Then $\forall s_i \in S$, $t^d_i = \frac{L^1}{v}$, and $\overline{D}^1 = \frac{\sum^N_{i=1} t^d_i}{N} = \frac{L^1}{v}$. Now, our target is to find a minimum cost route that starts from the sink, accesses all the sensors and goes back to the sink. This is the aim of the classical TSP. Now TSP is reduced to the *CWFRD* problem. Because TSP has been proved to be NP-hard [20], the CWFRD problem is also NP-hard.

■

4. Integer Linear Programming Formulation

In this section, the *Integer Linear Programming* (*ILP*) formulation of the *CWFRD* problem is given. Using this formulation, the *CWFRD* problem could be efficiently and optimally solved in a reasonable time using the existing *ILP* solving techniques or tools (such as CPLEX [21] and LINGO [22]) when the node number N is small.

Let S_m denote the subset of S, then $\bigcup^M_{m=1} S_m = S$, where M is the number of the subsets. We use N_m to denote the node number in S_m. s^m_0 and $s^m_{N_m+1}$ in S_m both represent the sink node s_0. We use V to denote the set $\{0, 1, 2, ..., N\}$. V' denotes the set $\{1, 2, ..., N\}$. Let x_{mij} be a 0-1 integer variable for each edge $[s_i \to s_j]$ such that $x_{mij} = 1$ if $[s_i \to s_j]$ is included in the sub-route R_m, and 0 otherwise. Let u_{mi} be a positive integer variable for each node s_i in sub-route S_m, showing the order in which the sensors are visited in the scheduled sub-route R_m.

Let L_m denote the total route length of sub-route R_m, i.e.,

$$L_m = \sum^{N_m}_{j=0} L(s_j, s_{j+1}). \tag{6}$$

And let W_m denote the total data size of the nodes in R_m, i.e.,

$$W_m = \sum^{N_m}_{i=1} w^m_i, \tag{7}$$

where w^m_i is the data size of the i^{th} node of R_m.

Then the weighted delay of R_m is

$$D_m = \frac{W_m \sum^m_{i=1} L_i}{v}, \tag{8}$$

Then the average weighted delay could be represented by

$$\overline{D} = \frac{\sum_{m=1}^{M} (W_m \sum_{i=1}^{m} L_i)}{v \sum_{m=1}^{M} W_m} = \frac{\sum_{m=1}^{M} (W_m \sum_{i=1}^{m} L_i)}{v \sum_{m=1}^{N} w_m} \tag{9}$$

The programming formulation of L_i, W_i and D_i are

$$L_m = \sum_{i \in V} \sum_{\substack{j \in V \\ j \neq i}} x_{mij} L(s_i, s_j), \tag{10}$$

$$W_m = \sum_{i \in V} \sum_{\substack{j \in V \\ j \neq i}} x_{mij} w_j, \tag{11}$$

and

$$D_m = \frac{\sum\limits_{i \in V} \sum\limits_{\substack{j \in V \\ j \neq i}} \left(x_{mij} w_j \left(\sum\limits_{m'=1}^{m} \sum\limits_{i' \in V} \sum\limits_{\substack{j \in V \\ j' \neq i}} x_{m'i'j'} L(s_{i'}, s_{j'}) \right) \right)}{v}. \tag{12}$$

Since each edge connects two vertices (sensors), we only consider the data of the end vertex (sensor) of each edge (i.e., w_j in Eq. (11)). Then when we consider all the edges in a sub-route, all the data of each sensor will be added only once.

Because the extreme case is that there is only one sensor in a sub-route, i.e., the subset number is equal to the node number, then $M = N$ in the formulation. Now the average weighted delay can be represented as:

$$\overline{D} = \frac{\sum\limits_{m=1}^{N} \sum\limits_{i \in V} \sum\limits_{\substack{j \in V \\ j \neq i}} \left(x_{mij} w_j \left(\sum\limits_{m'=1}^{m} \sum\limits_{i' \in V} \sum\limits_{\substack{j' \in V \\ j' \neq i'}} x_{m'i'j'} L(s_{i'}, s_{j'}) \right) \right)}{v \sum\limits_{i \in V'} w_i} \tag{13}$$

Since $v \sum_{i=1}^{N} w_i$ is a constant value, then the objective function could be

$$\sum_{m \in V} \sum_{i \in V} \sum_{\substack{j \in V \\ j \neq i}} \sum_{m'=1}^{m} \sum_{i' \in V} \sum_{\substack{j' \in V \\ j' \neq i'}} x_{mij} x_{m'i'j'} w_j L(s_{i'}, s_{j'}) \tag{14}$$

When the value of Eq. (14) is minimum, the value of Eq. (13) must also be minimum. It is worth noting that, this objective function is an non-linear function, because it contains the term $x_{mij}x_{m'i'j'}$, which is quadratic. Because x_{mij} and $x_{m'i'j'}$ are 0-1 integers, $x_{mij}x_{m'i'j'}$ is also a 0-1 integer. Now we introduce a new 0-1 integer variable $x_{mijm'i'j'}$ to replace $x_{mij}x_{m'i'j'}$, so as to transform the quadratic objective function to a linear function. And some constraints should be added to make sure that $x_{mijm'i'j'} \equiv x_{mij}x_{m'i'j'}$ for all the combinations of x_{mij} and $x_{m'i'j'}$.

The *ILP* formulation of the CWFRD problem can thus be stated as follows:

ILP Objective Function:

$$\sum_{m \in V} \sum_{i \in V} \sum_{\substack{j \in V \\ j \neq i}} \sum_{m'=1}^{m} \sum_{i' \in V} \sum_{\substack{j' \in V \\ j' \neq i'}} x_{mijm'i'j'} w_j L(s_{i'}, s_{j'}) \tag{15}$$

ILP Constraints:

$$\begin{cases} \sum_{m\in V'}\sum_{\substack{j\in V\\ j\neq i}} x_{mij} = 1, & \forall i \in V' \quad (16)\\ \sum_{m\in V'}\sum_{\substack{i\in V\\ i\neq j}} x_{mij} = 1, & \forall j \in V' \quad (17)\\ \sum_{i\in V} x_{mij} - \sum_{i\in V} x_{mji} = 0, & \forall m, j \in V' \quad (18)\\ u_{mi} - u_{mj} + N x_{mij} \leq N-1, & \forall m, i, j \in V' \quad (19)\\ 2x_{mijm'i'j'} - x_{mij} - x_{m'i'j'} \leq 0, & \forall m' \leq m \in V', \forall i, j, i', j' \in V \quad (20)\\ x_{mij} + x_{m'i'j'} - 2x_{mijm'i'j'} \leq 1, & \forall m' \leq m \in V', \forall i, j, i', j' \in V \quad (21) \end{cases}$$

Equations (16) and (17) represent the conditions that each node (other than s_0) is visited exactly once, where Eq. (16) ensures that each node (other than s_0) has only one former node, and Eq. (17) guarantees that each node (other than s_0) has only one successive node. Equation (18) ensures that each node is assigned to only one sub-route, which aims to avoid the discontinuous sub-route (e.g., $[0 \rightarrow 1]$ and $[2 \rightarrow 0]$ belong to sub-route R_1, but $[1 \rightarrow 2]$ belongs to sub-route R_2). Equation (19) is called the Miller-Tucker-Zemlin (MTZ) formulation [23] of the TSP, which aims to eliminate routes that do not begin and end at s_0 in each sub-route. The details and the proof of Eq. (19) can be found in [23]. According to the following theorem, Eqs. (20) and (21) can guarantee that the non-linear objective function (Eq. (14)) and the linear one (Eq. (15)) are equivalent.

Theorem 2: The quadratic function (Eq. (14)) is equivalent to the linear function (Eq. (15)) under the constraints of Eqs. (20) and (21).

Proof: We will show under all the combinations of x_{mij} and $x_{m'i'j'}$, $x_{mijm'i'j'} \equiv x_{mij}x_{m'i'j'}$.

1) $x_{mij} = x_{m'i'j'} = 0$, then $x_{mij}x_{m'i'j'} = 0$.
   According to Eqs. (20) and (21) we have $x_{mijm'i'j'} \leq 0$ and $x_{mijm'i'j'} \geq -\frac{1}{2}$. Because $x_{mijm'i'j'}$ is a 0-1 integer (i.e., $x_{mijm'i'j'} = 0$ or $x_{mijm'i'j'} = 1$), $x_{mijm'i'j'} = 0$. Then $x_{mijm'i'j'} = x_{mij}x_{m'i'j'}$.
2) $x_{mij} = 0$ and $x_{m'i'j'} = 1$ (or $x_{mij} = 1$ and $x_{m'i'j'} = 0$), then $x_{mij}x_{m'i'j'} = 0$.
   According to Eqs. (20) and (21) we have $x_{mijm'i'j'} \leq \frac{1}{2}$ and $x_{mijm'i'j'} \geq 0$. Because $x_{mijm'i'j'}$ is a 0-1 integer (i.e., $x_{mijm'i'j'} = 0$ or $x_{mijm'i'j'} = 1$), $x_{mijm'i'j'} = 0$. Then $x_{mijm'i'j'} = x_{mij}x_{m'i'j'}$.
3) $x_{mij} = x_{m'i'j'} = 1$, then $x_{mij}x_{m'i'j'} = 1$.
   According to Eqs. (20) and (21) we have $x_{mijm'i'j'} \leq 1$ and $x_{mijm'i'j'} \geq 1$, then $x_{mijm'i'j'} = 1$. Thus $x_{mijm'i'j'} = x_{mij}x_{m'i'j'}$.

Then $x_{mijm'i'j'} \equiv x_{mij}x_{m'i'j'}$ under all the combinations of x_{mij} and $x_{m'i'j'}$. And thus Eq. (14) is equivalent to Eq. (15).

■

This *ILP* objective function involves $N(N+1)^2(N^3+1)$ variables, where the number of x_{mij} is $N^2(N+1)$, the number of u_{mi} is $N(N+1)$, and the number of $x_{mijm'i'j'}$ is

$N^4(N+1)^2$. Equations (16)-(21) involve N, N, N^2, $N^2(N-1)$, $\frac{N^4(N+1)^2}{2}$ and $\frac{N^4(N+1)^2}{2}$ constraints, respectively. Then the total number of the constraints is $N^4(N+1)^2 + N^3 + 3N$. With the increase of the node number N, the total number of variables and constraints for the *ILP* would be greatly increased. It may not be computationally practical to solve the *ILP*. Nevertheless, the *ILP* formulation is presented here to provide some insight into the *CWFRD* problem.

5. A Heuristic Algorithm

As the *CWFRD* problem has been proved to be NP-hard in section 3, there are no fast optimal solutions for this problem. In this section, we will propose an efficient heuristic algorithm to solve it.

The method to solve *CWFRD* is shown in **Algorithm 1**. We call this method the *Initialization-Split-Optimization* algorithm (or *ISO* for short). In this algorithm, we first generate an initial route. Then we split the initial route into sub-routes, which could reduce the average weighted delay. Finally, the sequence of the sub-routes is refined using the local optimization method. Now we will discuss how to implement these three sub-algorithms of *ISO*.

Algorithm 1 *ISO()*

Input: s_0, S
Output: $\mathbb{R}_{ISO}$;
1: $\mathbb{R}_{RI} \leftarrow RouteInitialization(s_0, S)$;
2: $\mathbb{R}_{RS} \leftarrow RouteSplit(\mathbb{R}_{RI})$;
3: $\mathbb{R}_{ISO} \leftarrow LocalOptimization(\mathbb{R}_{RS})$;
4: **return** $\mathbb{R}_{ISO}$;

5.1 Route Initialization Methods

We will first show how to get an initial route when the scenario is given. Clearly, the average weighted delay is related to both the data size and the route length. Then we will propose three heuristic route initialization algorithms considering different metrics based on the data size and the route length.

5.1.1 Weightiest Data First Algorithm

The *Weightiest_Data_First (WDF)* algorithm would seem to be the first choice to consider, in which the node with the weightiest data size is selected as the next node. The algorithm is shown in **Algorithm 2**.

Algorithm 2 *Weightiest_Data_First()*

Input: s_0, S
Output: $\mathbb{R}_{WDF}$;
1: $\mathbb{R}_{WDF} \leftarrow$ Sort S by descendant order of w_i;
2: $\mathbb{R}_{WDF} \leftarrow [s_0 \rightarrow \mathbb{R}_{WDF} \rightarrow s_0]$;
3: **return** $\mathbb{R}_{WDF}$;

5.1.2 Shortest Path Algorithm

Now we consider the route length and the *Shortest_Path* (*SP*) algorithm is proposed, which is shown in **Algorithm 3**. In this algorithm, the initial route is calculated using an existing TSP solution, which is denoted by *TSP* in line 1.

Algorithm 3 *Shortest_Path()*

Input: s_0, S

Output: $\mathbb{R}_{SP}$;

1: $\mathbb{R}_{SP} \leftarrow TSP(s_0, S)$;

2: **return** $\mathbb{R}_{SP}$;

5.1.3 Weighted Nearest Neighbor Algorithm

The *Weighted_Nearest_Neighbor (WNN)* algorithm would seem to be another choice to be taken into consideration, which considers both the data size and the route length, and the weighted nearest neighbor of the current node is selected as the next node. This algorithm is shown in **Algorithm 4**. In this algorithm, $\mathbb{R}_{WNN}$ denotes the scheduled sequence of the sensors, $R_{current}$ denotes the current subset, and $s_{current}$ denotes the newly-selected sensor. The function $W(s')$ in line 5 denotes the data size of node s'. S_{remain} represents the remaining sensors which have not been scheduled.

Algorithm 4 *Weighted_Nearest_Neighbor()*

Input: s_0, S

Output: $\mathbb{R}_{WNN}$;

1: $\mathbb{R}_{WNN} \leftarrow [s_0]$;

2: $S_{remain} \leftarrow S$;

3: $s_{current} \leftarrow s_0$;

4: **while** $S_{remain} != \emptyset$ **do**

5: find the node s' in S_{remain} that the value of $\frac{W(s')}{L(s_{current}, s')}$ is the maximum;

6: $S_{remain} \leftarrow S_{remain} - s'$;

7: $s_{current} \leftarrow s'$;

8: $\mathbb{R}_{WNN} \leftarrow [\mathbb{R}_{WNN} \rightarrow s_{current}]$;

9: **end while**

10: $\mathbb{R}_{WNN} \leftarrow [\mathbb{R}_{WNN} \rightarrow s_0]$;

11: **return** $\mathbb{R}_{WNN}$;

5.2 Route Split Method

After getting an initial route using the route initialization methods proposed in last subsection, we will split this initial route into sub-routes, i.e., deciding between which two sensors to insert a sink. The $Route\_Split$ algorithm is shown in **Algorithm 5**. In this algorithm, we will sequentially test the all the possible locations between two adjacent sensors to insert the sink s_0. If the average weighted delay will be reduced when we insert s_0 between two sensors, then s_0 is inserted here. The sensors accessed before the new inserted s_0 are removed, and we only consider the remaining nodes in the route. This is controlled by N_{start}. $\mathbb{R}$ is the initialized route, and the first and last node in $\mathbb{R}$ are both s_0. Then the length of $\mathbb{R}$ is $N+2$. We repeat the procedure until there is no sensor left. The function *TSP* in lines 9 and 10 denotes the TSP solution, and the function *AWD* in line 11 returns the average weighted delay of a route. $\mathbb{R}(a:b)$ returns a subset of the nodes between the a^{th} node and the b^{th} node of $\mathbb{R}$, e.g., $\mathbb{R}(1:1)$ in line 2 denotes the first node in route $\mathbb{R}$.

Algorithm 5 *Route_Split()*

Input: $\mathbb{R}$

Output: $\mathbb{R}_{RS}$;

$\mathbb{R}_{RS} \leftarrow \emptyset$;
$s_0 \leftarrow \mathbb{R}(1:1)$;
$N \leftarrow$ the node number in $\mathbb{R}$ except s_0;
$N_{start} \leftarrow 2$;
$N_{split} \leftarrow N_{start}$;
while $N_{split} \leq N$ **do**
  $\mathbb{R}_{old} \leftarrow [s_0 \rightarrow \mathbb{R}(N_{start} : N+1) \rightarrow s_0]$;
  $\mathbb{R}_{new} \leftarrow [s_0 \rightarrow \mathbb{R}(N_{start} : N_{split}) \rightarrow s_0 \rightarrow \mathbb{R}(N_{split}+1 : N+1) \rightarrow s_0]$;
  $\mathbb{R}_{old\_TSP} \leftarrow TSP(\mathbb{R}_{old})$;
  $\mathbb{R}_{new\_TSP} \leftarrow TSP(\mathbb{R}_{new})$;
  if $AWD(\mathbb{R}_{new\_TSP}) < AWD(\mathbb{R}_{old\_TSP})$ **then**
    $\mathbb{R}'_{new} \leftarrow [s_0 \rightarrow \mathbb{R}(N_{start} : N_{split}) \rightarrow s_0]$;
    $\mathbb{R}'_{new\_TSP} \leftarrow TSP(\mathbb{R}'_{new})$;
    $\mathbb{R}_{RS} \leftarrow [\mathbb{R}_{RS} \rightarrow \mathbb{R}'_{new\_TSP}]$;
    $N_{start} \leftarrow N_{split} + 1$;
  end if
  $N_{split} \leftarrow N_{split} + 1$;
end while
return $\mathbb{R}_{RS}$;

5.3 Local Optimization Method

The *Local_Optimization* algorithm is shown in **Algorithm 6**. In this algorithm, the sub-routes are sorted by ascendant order of $\lambda_m = \frac{L_m}{W_m}$. We will show that the result of this algorithm is optimal.

Algorithm 6 *Local_Optimization()*

Input: $\mathbb{R}$

Output: $\mathbb{R}_{LO}$;

for all sub-route R_m in $\mathbb{R}$ **do**
  $L_m \leftarrow L(R_m)$;
  $W_m \leftarrow W(R_m)$;
  $\lambda_i \leftarrow \frac{L_m}{W_m}$;
end for
$\mathbb{R}_{LO} \leftarrow$ sort the sub-routes by ascendant order of λ_m;
return $\mathbb{R}_{LO}$;

Let route $\mathbb{R}_1 = [R_1 \rightarrow \cdots \rightarrow R_{k-1} \rightarrow R_k \rightarrow R_{k+1} \rightarrow R_{k+2} \rightarrow \cdots \rightarrow R_M]$ and $\mathbb{R}_2 = [R_1 \rightarrow \cdots \rightarrow R_{k-1} \rightarrow R_{k+1} \rightarrow R_k \rightarrow R_{k+2} \rightarrow \cdots \rightarrow R_M]$, the average weighted delay of $\mathbb{R}_1$ and $\mathbb{R}_2$ are denoted by $\overline{D}_1$ and $\overline{D}_2$ respectively. Then we have the following lemma,

Lemma 1: If $\lambda_k > \lambda_{k+1}$, then $\overline{D}_1 - \overline{D}_2 > 0$.

Proof: According to Eq. (9), the average weighted delay of $\mathbb{R}_1$ is

$$\overline{D}_1 = \frac{\sum_{i=1}^{M}\left(W_i \sum_{j=1}^{i} L_j\right)}{v\sum_{i=1}^{M} W_i}$$
$$= \frac{\sum_{i=1}^{k-1}\left(W_i \sum_{j=1}^{i} L_j\right) + W_k\left(L_k + \sum_{j=1}^{k-1} L_j\right) + W_{k+1}\sum_{j=1}^{k+1} L_j + \sum_{i=k+2}^{M}\left(W_i \sum_{j=1}^{i} L_j\right)}{v\sum_{i=1}^{M} W_i}.$$

And the average weighted delay of $\mathbb{R}_2$ is

$$\overline{D}_2 = \frac{\sum_{i=1}^{k-1}\left(W_i \sum_{j=1}^{i} L_j\right) + W_{k+1}\left(L_{k+1} + \sum_{j=1}^{k-1} L_j\right) + W_k\sum_{j=1}^{k+1} L_j + \sum_{i=k+2}^{M}\left(W_i \sum_{j=1}^{i} L_j\right)}{v\sum_{i=1}^{M} W_i}$$

Then

$$\overline{D}_1 - \overline{D}_2 = \frac{W_k\left(L_k + \sum_{j=1}^{k-1} L_j\right) + W_{k+1}\sum_{j=1}^{k+1} L_j - W_{k+1}\left(L_{k+1} + \sum_{j=1}^{k-1} L_j\right) - W_k\sum_{j=1}^{k+1} L_j}{v\sum_{i=1}^{M} W_i}$$
$$= \frac{W_kL_k + W_k\sum_{j=1}^{k-1} L_j + W_{k+1}\sum_{j=1}^{k+1} L_j - W_{k+1}L_{k+1} - W_{k+1}\sum_{j=1}^{k-1} L_j - W_k\sum_{j=1}^{k+1} L_j}{v\sum_{i=1}^{M} W_i}$$
$$= \frac{W_kL_k - W_{k+1}L_{k+1} - W_k(L_k + L_{k+1}) + W_{k+1}(L_k + L_{k+1})}{v\sum_{i=1}^{M} W_i}$$
$$= \frac{W_{k+1}L_k - W_kL_{k+1}}{v\sum_{i=1}^{M} W_i}$$

Since $\lambda_k > \lambda_{k+1}$, i.e., $\frac{L_k}{W_k} > \frac{L_{k+1}}{W_{k+1}}$, then $W_{k+1}L_k > W_kL_{k+1}$. And thus $\overline{D}_1 - \overline{D}_2 > 0$.

■

Theorem 3: The sub-route sequence generated by **Algorithm 6** has the minimum average weighted delay.

Proof: Assume that $\mathbb{R}$ is the route with the minimum average weighted delay. If the sub-routes in $\mathbb{R}$ is not sequenced according to λ_i, then there must exist sub-routes R_k and R_{k+1} that satisfy $\lambda_k > \lambda_{k+1}$. According to **Lemma 1**, the average weighted delay could be reduced by swap R_k and R_{k+1}, which conflicts with that $\mathbb{R}$ is the route with the minimum average weighted delay.

■

5.4 Complexity Analysis

Assume that *N* denotes the number of the sensor nodes. The complexity of the *ISO* algorithm denpends heavily on that of the TSP solution adopted in *ISO*. Let *O(M)* denote the complexity of *TSP*. Then the complexity for each step of the *ISO* algorithm is as follows:

- *Initialization Step:*
  - *WDF: WDF* is dominated by sorting of the nodes with respect to the weight of the data resulting in *O(NlogN)* complexity.

- *SP:* Since the complexity of *SP* is equal to that of *TSP*, then the complexity of *SP* is also *O(M)*.
- *WNN*: The complexity of line 5 is *O(N)*, and this line will be repeated for *N* times. And thus the complexity of *WNN* is *O(N*^{2}*)*.

- *Split Step: TSP* in the while loop (lines 6-18 of **Algorithm 5**) will be repeated for *N* times, so the complexity of *Route_Split* algorithm is *O(NM)*.
- *Optimization Step: Local_Optimization* is dominated by the sort operation in line 6 of Algorithm 6, and the complexity is *O(NlogN)*.

Since existing heuristic algorithms for *TSP* have complexity of *O(N*^{2}*)* or higher [24,25], i.e., $M \geq N^2$, then *ISO* has an overall time complexity of *O(NM)*, which is dominated by the *Route_Split* algorithm. Although the time complexities of the three algorithms for initialization are not the same, they do not affect the overall time complexity of *ISO*.

6. Performance Evaluation

In this section, we will evaluate the performances of the proposed algorithms through simulation. In our experiments, the following settings are used. The sensors are randomly distributed in a square of $1000 \times 1000\ m^2$. The speed of the ferry is set to $5\ m/s$. The data size of each sensor is uniformly chosen from [1 100] (unit). We simulate two scenarios of the sink location. The first scenario is that the sink is located at the center of the simulation area, i.e., the coordinate of s_0 is $(500, 500)m$, and we refer this scenario as the *center scenario* in the experiment. The other scenario is that the sink is located at the corner of the simulation area, i.e., the coordinate of s_0 is $(0, 0)m$, and we refer this scenario as the *corner scenario*. We repeat the simulations for 1000 times. The TSP solution used in the experiment is LKH [24], which is one of the state of the art fast heuristic algorithms for TSP.

Because we have proposed three route initialization algorithms (i.e., the *Weightiest_Data_First*, *Shortest_Path* and *Weighted_Nearest_Neighbor* algorithms), the *ISO* algorithm will also have three versions using these three methods as the *RouteInitialization* algorithm. In this experiment, the three versions of the *ISO* algorithms are referred as *WDF*, *SP* and *WNN*, which are the abbreviations of the three *RouteInitialization* algorithms used in *ISO*, respectively. For comparison, we also show the result of the simple cycle route algorithm, which adopts the classic TSP algorithm (LKH in the experiment). We refer this algorithm as *CYCLE* in the figures.

The average weighted delay and route length will be chosen as the performance metrics of the algorithms to be evaluated in this paper. Route length is the total length of all the simple cycles of a route. In order to compare the performance of the different algorithms more clearly, we use the relative performance as a new metric, which is defined as follows:

Definition 3 (***Relative Performance***): The relative performance of an algorithm $\mathcal{A}$ is defined as $\rho_{\mathcal{A}} = \frac{P(R_{\mathcal{A}})}{P(R_{CYCLE})}$, where $R_{\mathcal{A}}$ is the route scheduled by $\mathcal{A}$, and $P(R_{\mathcal{A}})$ denotes the performace of $R_{\mathcal{A}}$.

P could be average weighted delay or route length. Clearly, $\rho_{CYCLE} = 1$. We will consider the impact of node number on the performance, and the node number is set to 5, 10, 20, 40 and 80 in each experiment.

We will first show the number of subroutes (simple cycles) in the route of each algorithm, and the results are shown in **Fig. 2**. This figure shows that the route scheduled by *WNN* always contains more subroutes than others. And *WDF* will always generate the small number of subroutes. This illustration can help to understand the results of two performance metrics mentioned above, which will be shown bellow.

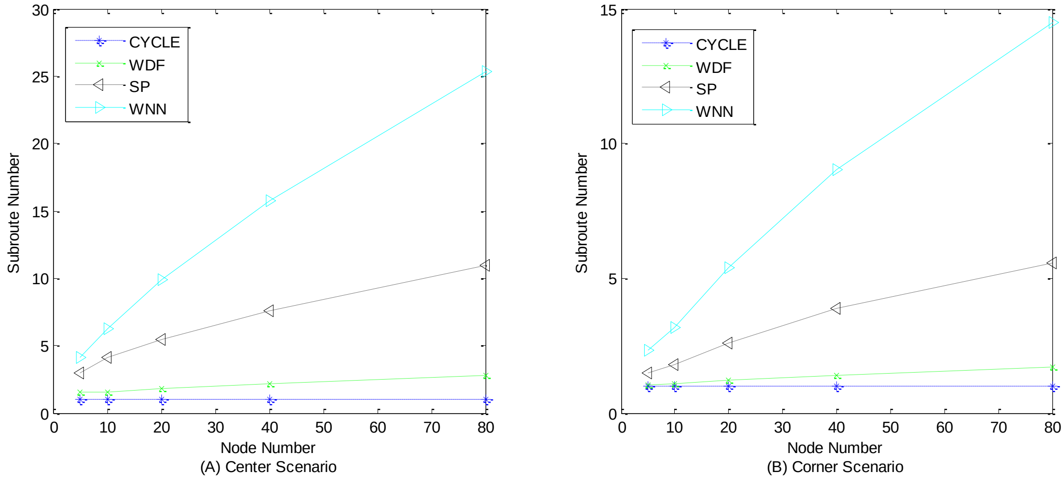

Fig. 2. Subroute Number VS. Node Number

6.1 Average Weighted Delay

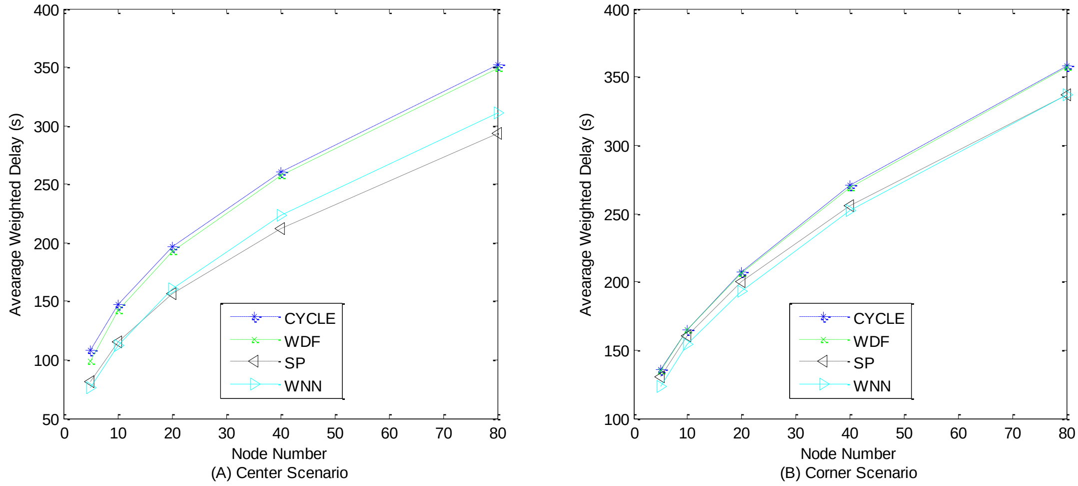

Fig. 3. Average Weighted Delay VS. Node Number

Fig. 3 shows the impact of node number on the average weighted delay under the center and corner scenario, respectively. And **Fig. 4** is the corresponding relative average weighted delay. From these two figures we can see that, the average weighted delay is increasing with the increase of the node number for all the algorithms under both the two scenarios. That is because when the node number is larger, the ferry needs to access more nodes and the route is longer, then the delay will be increased. The Average Weighted Delay of *CYCLE* is the highest of the four algorithms for all the node numbers under both the two scenarios. The reason is that there is only one simple cycle in *CYCLE*, and the message cannot be delivered until the ferry has accessed all the sensors. The performance of *SP* is better than *WDF* under both the two scenarios. And we can infer that the distance plays a more important role than the data size for the average weighted delay in *CWFRD*. **Fig. 4-(A)** shows that when the node number is small, the performance of *WNN* is the best under the center scenario. It can reduce about 30% of average weighted delay compared to *CYCLE* when the node number is 5. When the node number is greater than 20, the performance of *SP* is the best. **Fig. 4-(B)** shows that the average weighted delay of *WNN* is the lowest under the corner scenario. Overall, the average weighted delay of *ISO* under the center scenario is better than that under the corner scenario.

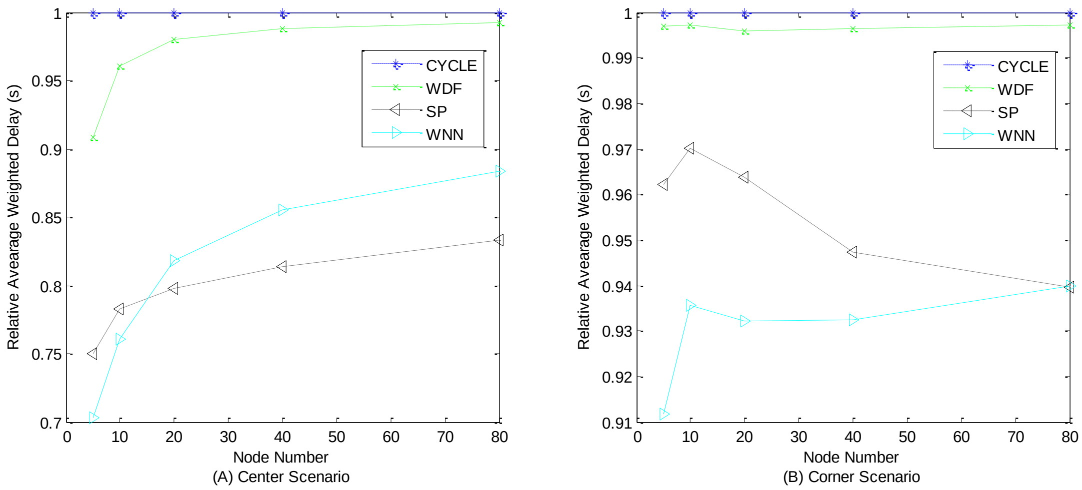

Fig. 4. Relative Average Weighted Delay VS. Node Number

6.2 Route Length

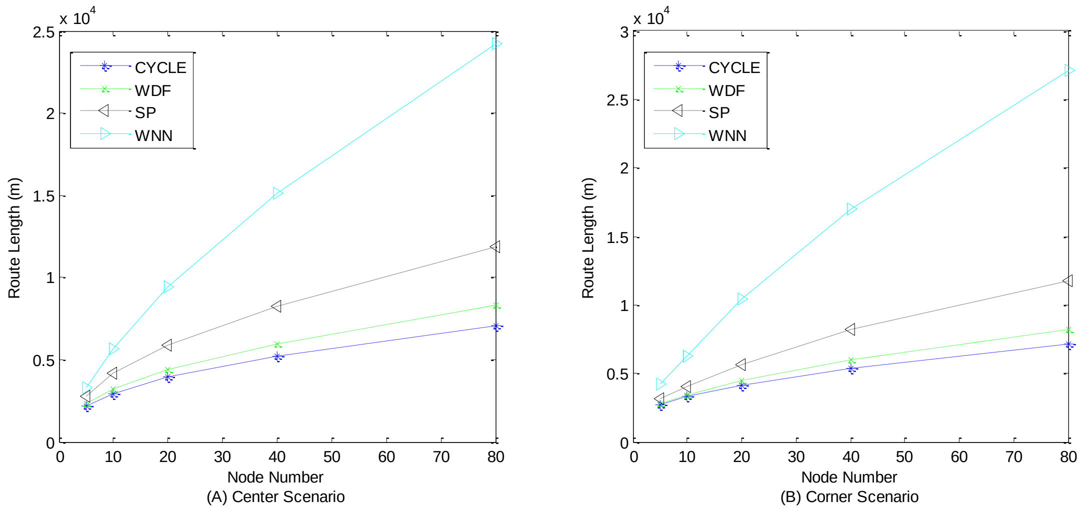

Fig. 5. Route Length VS. Node Number

Fig. 5 and **6** illustrate the impact of node number on the route length under the two different scenarios. There is no doubt that the route length of *CYCLE* is the shortest, since the objective of *CYCLE* is to minimize the route length. These two figures show that, with the increase of node number, the relative route lengths of *WDF*, *SP* and *WNN* are increasing. *WNN* increases the fastest of all the three *ISO* algorithms. The reason is that the route scheduled by *WNN* contains more subroutes than other algorithms (see **Fig. 2**), and the ferry needs to access the sink more often, which will surely lead to the extra cost in route length. When the node number is 80 under the corner scenario, the route length of *WNN* is more than 3.5 times of that of

CYCLE. The route lengths of the same algorithm under two different scenarios do not differ significantly.

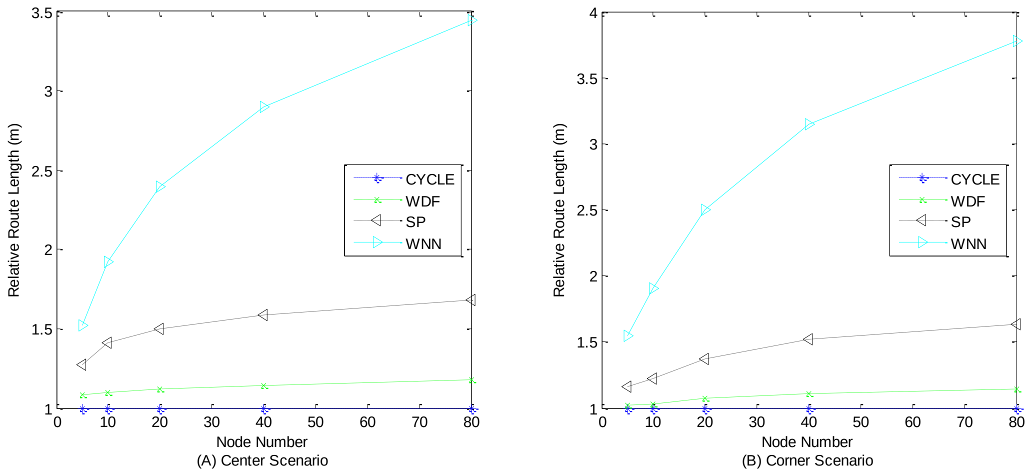

Fig. 6. Relative Route Length VS. Node Number

From the experimental results given above we can see that, *WNN* performs best in average weighted delay but worst in route length, and *CYCLE* performs worst in average weighted delay but best in route length. *SP* makes a good balance between performance (average weighted delay) and cost (route length). It achives even less average weighted delay than *WNN* for some scenarios (see **Fig. 4-(A)**), and performs much better than *WNN* in route length.

7. Conclusion

In this paper, the *CWFRD* problem is investigated. In *CWFRD*, the ferry route is designed as a closed walk rather than a simple cycle. The experiment results show that the average weighted delay can be significantly reduced by scheduling a closed walk route, compared to the simple cycle route. In the future work, the dynamic traffic should be taken into consideration.

Acknowledgment

This work has been partially supported by the National Natural Science Foundation of China under Grant No. 61272139, National Science and Technology Major Projects of China under Grant No. 2013ZX01028-001-002.

References

[1] A. S. Arun, R. Aditya, and B. S. Mani, “Mobile element scheduling with dynamic deadlines,” *Mobile Computing, IEEE Transactions on*, vol. 6, no. 4, pp. 395–410, 2007. Article (CrossRef Link).

[2] P. Juang, H. Oki, Y. Wang, M. Martonosi, L. S. Peh, and D. Rubenstein, “Energy-efficient computing for wildlife tracking: design tradeoffs and early experiences with zebranet,” *SIGOPS Oper. Syst. Rev.*, vol. 36, no. 5, pp. 96–107, 2002. Article (CrossRef Link).

[3] R. Szewczyk, E. Osterweil, J. Polastre, M. Hamilton, A. Mainwaring, and D. Estrin, "Habitat monitoring with sensor networks," *Commun. ACM*, vol. 47, pp. 34–40, June 2004. Article (CrossRef Link).

[4] A. Vahdat and D. Becker, "Epidemic routing for partially connected ad hoc networks," *tech. rep.*, Duke University, 2000. Article (CrossRef Link).

[5] W. Zhao and M. Ammar, "Message ferrying: Proactive routing in highly-paritioned wireless ad hoc networks," in *Proc. of the 9th IEEE International Workshop on Future Trends of Distributed Computing Systems*, (Puerto Rico), pp. 308–314, 2003. Article (CrossRef Link).

[6] W. Zhao, M. Ammar, and E. Zegura, "A message ferrying approach for data delivery in sparse mobile ad hoc networks," in *Proc. of the 5th ACM international symposium on Mobile ad hoc networking and computing*, (Roppongi Hills, Tokyo, Japan), pp. 187–198, ACM, 2004. Article (CrossRef Link).

[7] W. Zhao, M. Ammar, and E. Zegura, "Controlling the mobility of multiple data transport ferries in a delay-tolerant network," in *Proc. of INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, vol. 2, pp. 1407– 1418, 2005. Article (CrossRef Link).

[8] P. Wei, Z. Baokang, Y. Wanrong, W. Chunqing, and Y. Xiaorong, "Ferry route design with delay bounds in delay-tolerant networks," in *Proc. of Computer and Information Technology (CIT), 2010 IEEE 10th International Conference* on, pp. 281–288, 2010. Article (CrossRef Link).

[9] W. Xiaoyang, C. Yingwen, X. Ming, and P. Wei, "Tafr: A ttl-aware message ferry scheme in dtn," in *Proc. of Computational and Information Sciences (ICCIS), 2012 Fourth International Conference* on, pp. 1380–1383, 2012. Article (CrossRef Link).

[10] R. C. Shah, S. Roy, S. Jain, and W. Brunette, "Data mules: modeling a three-tier architecture for sparse sensor networks," in *Proc. of Sensor Network Protocols and Applications, Proceedings of the First IEEE.*, pp. 30– 41, 2003. Article (CrossRef Link).

[11] Y. Gu, D. Bozdag, E. Ekici, F. Ozguner, and C.-G. Lee, "Partitioning based mobile element scheduling in wireless sensor networks," in *Proc. of Sensor and Ad Hoc Communications and Networks (SECON), Second Annual IEEE Communications Society Conference* on, pp. 386 – 395, sept., 2005. Article (CrossRef Link).

[12] L. Tong, Q. Zhao, and S. Adireddy, "Sensor networks with mobile agents," in *Proc. of 2003 Military Communications Intl Symp*, pp. 688–693, 2003. Article (CrossRef Link).

[13] J. Chen, Y. Chen, L. Zhou, and Y. Du, "A data gathering approach for wireless sensor network with quadrotor-based mobile sink node," *KSII Transactions on Internet and Information Systems*, vol. 6, no. 10, pp. 2529–2547, 2012. Article (CrossRef Link).

[14] M. M. B. Tariq, M. A. Zegura, and E. Zegura, "Message ferry route design for sparse ad hoc networks with mobile nodes," 2006. Article (CrossRef Link).

[15] A. A. Somasundara, A. Ramamoorthy, and M. B. Srivastava, "Mobile element scheduling for efficient data collection in wireless sensor networks with dynamic deadlines," *in Proc. of Real-Time Systems Symposium*, 2004. *Proceedings. 25th IEEE International, Real-Time Systems Symposium*, 2004. Proceedings. 25th IEEE International, pp. 296– 305, 2004. Article (CrossRef Link).

[16] T. Wang and C. P. Low, "The general message ferry route (mfr*) problem and the an-improved-route (air) scheme," *Comput. Netw.*, vol. 56, no. 4, pp. 1442–1457, 2012. Article (CrossRef Link).

[17] J. L. Bentley, "Fast algorithms for geometric traveling salesman problems," *Informs Journal on Computing*, vol. 4, pp. 387–411, 1992. Article (CrossRef Link).

[18] S. Arora, "Polynomial time approximation schemes for euclidean traveling salesman and other geometric problems," *J. ACM*, vol. 45, pp. 753–782, Sept. 1998. Article (CrossRef Link).

[19] R. Diestel, Graph Theory, vol. 173 of *Graduate Texts in Mathematics*. Springer-Verlag, Heidelberg, third ed., 2005. Article (CrossRef Link).

[20] E. Lawler, The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization. *Wiley-Interscience series in discrete mathematics*, Wiley, 1987. Article (CrossRef Link).

[21] CPLEX.http://www-01.ibm.com/software/integration/optimization/cplex-optimizer/. 2013.Article (CrossRef Link).
[22] LONGO.http://www.lindo.com/products/lingo/. 2013. Article (CrossRef Link).
[23] C. E. Miller, A. W. Tucker, and R. A. Zemlin, "Integer programming formulation of traveling salesman problems," *J. ACM*, vol. 7, no. 4, pp. 326–329, 1960. Article (CrossRef Link).
[24] LKH.http://www.akira.ruc.dk/~keld/research/LKH/.Article (CrossRef Link).
[25] T. H. Cormen, C. E. Leiserson, R. L. Rivest. *Introduction to algorithms*. The MIT press, 2009. Article (CrossRef Link).

Qiang Dou received the B.Sc. and the Ph.D. degrees in Computer Science from the National University of Defense Technology, China, in 1995 and 2001, respectively. Now he is a research fellow in College of Computer, National University of Defense Technology, China. His major interests are high performance computer architecture, high performance microprocessor design and verification, and computer networks.

Yong Wang received the B.Sc. and M.Sc. degrees from National University of Defense Technology, China, in 2006 and 2008, respectively. He is currently a Ph.D. student in the College of Computer, National University of Defense Technology under the supervision of Professor Zhenghu Gong. He was an exchange student at the Department of Computing, The Hong Kong Polytechnic University from May 2011 to April 2012. His current research interests include delay tolerant network, and mobile wireless network.

Wei Peng received the M.Sc. and the Ph.D. degrees in Computer Science from the National University of Defense Technology, China, in 1997 and 2000, respectively. From 2001 to 2002, he was a research assistant in College of Computer, National University of Defense Technology, China, then a research fellow from 2003. He was a visiting scholar of University of Essex, UK, from May 2007 to April 2008. His major interests are internet routing, network science, and mobile wireless networks.

Zhenghu Gong is a professor in National University of Defense Technology. His research interests include wireless network, and network communication and protocol.