Communication Cost Minimization in Wireless Sensor and Actor Networks for Road Surveillance

Wenzhong Li, *Member, IEEE*, Edward Chan, *Member, IEEE*, Mounir Hamdi, *Senior Member, IEEE*, Sanglu Lu, *Member, IEEE*, and Daoxu Chen, *Member, IEEE*

***Abstract*—In recent years, wireless sensor and actor networks (WSANs) have been extensively deployed to monitor physical environment and facilitate decision making based on data collected. Emerging applications such as road surveillance highlight some interesting research issues in WSANs, including coordination problems in sensor–actor or actor–actor communications. In this paper, the issue of choosing a set of working actors for coordinating data transmission in a road sensor and actor network with minimum communication cost is studied. A theoretical model is introduced to analyze the communication cost of data transmission in WSANs, and the sensor–actor coordination problem is formulated as an optimization problem. It is demonstrated that the problem can be divided into subproblems, and optimal solutions can be obtained by using a dynamic programming algorithm. A novel graph-based algorithm is also proposed with a communication-cost graph used to depict the cost of data transmission and a modified Dijkstra's algorithm to find optimal solutions in reduced time complexity. The efficiency of the proposed algorithms is confirmed using extensive simulations.**

***Index Terms*—Communication optimization, road surveillance, sensor-actor coordination problem, sensor and actor network.**

I. Introduction

Wireless sensor and actor networks (WSANs), which are composed of a set of sensors and actors linked by wireless medium to perform distributed sensing and acting tasks, have been widely used in battlefield surveillance, chemical attack detection, home automation, and environmental monitoring [1]–[3]. Sensors are low-cost, low-power devices with limited sensing, computation, and wireless communication capacities. Actors are assumed to be equipped with better processing capabilities, higher transmission power, and longer battery life. In WSANs, sensors and actors work together in data-centric applications, with sensors gathering information about the physical world and actors taking appropriate actions on the environment [4].

Manuscript received September 1, 2009; revised February 3, 2010, May 10, 2010, and September 20, 2010; accepted November 2, 2010. Date of publication November 29, 2010; date of current version February 18, 2011. This work was supported in part by the National Natural Science Foundation of China under Grant 60803111, Grant 90718031, and Grant 60721002, by the 973 Program of China under Grant 2009CB320705, by the JiangSu Natural Science Foundation under Grant BK2009100, and by a grant from the City University of Hong Kong under Project 7002494. The review of this paper was coordinated by Prof. S. Ci.

W. Li, S. Lu, and D. Chen are with the Department of Computer Science and Technology, Nanjing University, Nanjing 210093, China (e-mail: lwz@nju.edu.cn; sanglu@nju.edu.cn; cdx@nju.edu.cn).

E. Chan is with the Department of Computer Science, City University of Hong Kong, Kowloon, Hong Kong (e-mail: csedchan@cityu.edu.hk).

M. Hamdi is with the Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Kowloon, Hong Kong (e-mail: hamdi@cse.ust.hk).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TVT.2010.2094631

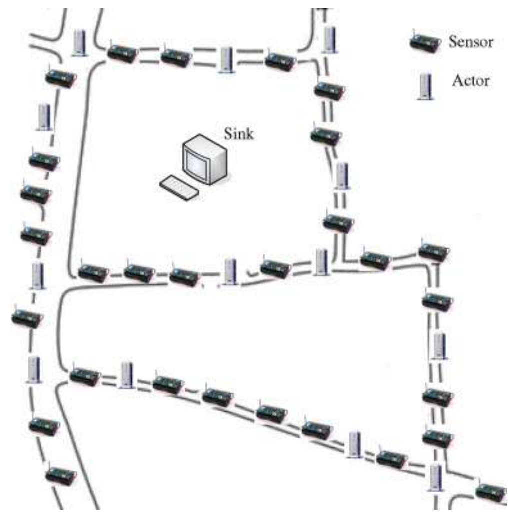

Fig. 1. Example of road surveillance.

Recently, researchers have proposed the use of WSANs for road surveillance; these are sometimes called road sensor and actor networks (RSANs) [5], [6]. Fig. 1 shows an example of an RSAN. A large number of sensors and actors are randomly deployed on public roads to collect real-time data on traffic and road conditions. When sensors detect an event (i.e., a passing car), they transmit sensing data (i.e., position and speed of the car) to nearby actors, which transmit data to the sink through long-range communication. The sink processes and analyzes the sensor data on board and then issues action commands to actors. Typical applications in RSANs include vehicle tracking, driver warning, and incident detection. For example, if a driver is going in the wrong direction, the sink will detect this and activate the appropriate actor to send a warning message to the driver. RSANs can also provide more sophisticated information services such as road-surface condition reporting and traffic information queries. There are a number of studies on RSANs, such as the use of sensor networks for vehicle tracking in military applications [7], [8]. Karpinski *et al.* proposed a software architecture for sensor networks in smart roads [6]. Jeong *et al.* presented an autonomous passive localization scheme for road sensor networks [5]. However, not much has been done to investigate how to achieve effective coordination in sensors and actors in RSANs.

While a large number of sensors and actors are typically deployed in a road network, such a dense deployment is usually not necessary for actor nodes since actors have higher capacities, and a single actor can cover a larger area. To reduce network overhead, some actors could be put to sleep to save energy and prolong network lifetime. The process of choosing a set of working sensors and actors and establishing suitable data paths between them is known as *sensor–actor coordination* [3], [4], [9]. Previous research has studied efficient data collecting, clustering, and sleeping schedule strategies in sensor networks [3], [10]–[17]. However, these strategies cannot be applied directly to RSANs because of the following: First, they work with different coordination patterns. The sensor–actor network is like a heterogeneous network combining powerful and resource-limited devices performing both long-range and short-range communications. Second, most existing clustering algorithms in sensor networks are topology dependent. In RSANs, the deployment of sensors and actors are restricted in a road network. This restriction in topology yields a different communication optimization problem, which will be discussed in this paper. To the best of our knowledge, the sensor–actor coordination issue with the explicit objective of minimizing network communication cost in road networks has not been studied in the literature.

In this paper, we address the following challenging research problem in the design of RSANs: how to choose a set of working actors and route sensing data between sensors and actors to minimize the total communication cost for road surveillance. Specifically, we make the following contributions.

1) We present a theoretical model to analyze the communication cost in an RSAN. By introducing the assumption of virtual actors deployed in road intersections, we formulate the problem of choosing the working actor set (WAS) for coordinating data transmission as an optimization problem.
2) We demonstrate that the optimization problem can be divided into subproblems; thus, a dynamic programming algorithm is proposed to obtain the optimal solution to achieve minimal communication cost. The time complexity of the dynamic programming algorithm is proven to be $O(nm^2)$, where n is the number of nodes, and m is the number of actors in RSANs.
3) We also introduce an efficient graph-based algorithm to solve the problem in lower time complexity. A *communication cost graph (CC-Graph)* is used to depict the cost of data transmission in RSANs, and a modified Dijkstra's algorithm is applied to find the shortest path, with an optimal solution to the problem obtained as the end result. The graph-based algorithm is proven to run in $O(nm + m^2 lgm)$, which is more efficient than the dynamic programming solution.
4) We further prove that, without the deployment of actors in virtual nodes, the proposed strategies produce a near-optimal solution, which approaches the optimal solution with asymptotic order $O(1/m)$.
5) We produce extensive simulations to study the system performance, which shows the efficiency of the proposed approaches.

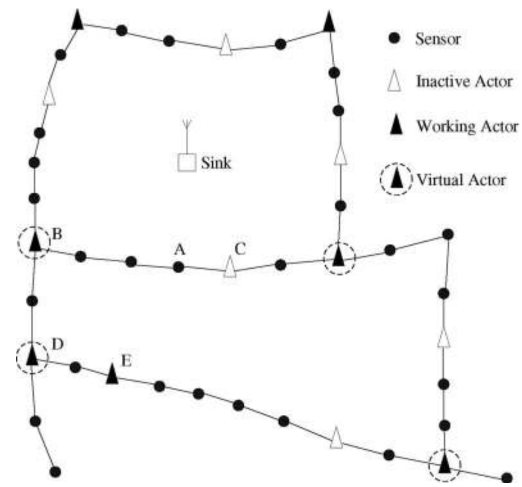

Fig. 2. RSAN.

The rest of this paper is organized as follows: Section II presents the system model of RSANs and formulates the communication cost minimization problem. Section III outlines both a dynamic programming solution and a graph-based solution to the problem and proves their correctness and optimality. The performance of the proposed algorithms is evaluated in Section IV. Section V introduces related work, and finally, Section VI concludes this paper.

II. System Model

Fig. 2 shows the road sensor–actor network that corresponds to Fig. 1. In this example, sensors detect events and send their data to nearby actors. Unlike other research assuming actors to be resource-rich nodes with unlimited power supply, we make much weaker assumptions about actors in our model: We only assume that actors are capable of sensing and performing long-range communications. That is, in our model, actors are not necessary to be powerful nodes; they could be resource-limited nodes operating on batteries, or they could be just normal sensors that are chosen to collect data and send them to the sink (like cluster heads). The assumption of coexistence of powerful and resource-limited actors coincides with our real life experience. For example, people tend to deploy powerful nodes in the busy roads to monitor their traffic condition. On the other hand, the rural area may not be covered by powerful actors, where resource-limited nodes will be used for long-range communication. In such a case, minimizing communication cost in the system is important to save energy and extend network lifetime.

Each actor has two states: *working* or *inactive*. If an actor is in the *working* state, it can sense events, collect data from nearby sensors, and establish long-range communication with the sink. If it is in the *inactive* state, it acts like a normal sensor. Consider the example in Fig. 2; if sensor A has detected an event, the data collected by the sensor need to travel three hops to send to the nearest actor B. However, if actor C is in the *working* state, it only needs one hop for A to communicate with C. More working actors clearly improves communication efficiency as the sensor data can be more quickly processed but

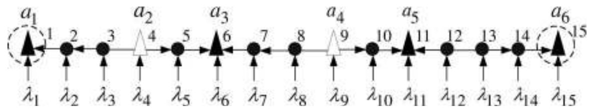

Fig. 3. Data communication in a road segment.

at the expense of increased communication cost and greater power consumption in the actors. Hence, the selection of a suitable set of working actors to minimize the total network communication cost is a key design decision in RSANs.

In most cases, RSANs are complicated and irregular networks, which are hard to analyze. To simplify the system model, we introduce *virtual actors*. We assume that there are virtual actors deployed at intersections of the road networks, and they are always in the working state. As shown in Fig. 2, the actors with dotted circle are virtual actors. The reason behind virtual nodes is that traffic is busier at the intersections and requires careful surveillance; hence, people tend to deploy actors in these places. In a real road, if no actors are deployed in such areas, the virtual node will redirect the workload to a nearby working actor. For example, in Fig. 2, node D is a virtual actor; if there is no real actor deployed in its area, the virtual node will direct its workload to a nearby working actor E or B.

The virtual nodes divide the road networks into a number of segments. Since virtual actors are assumed to always stay in the working state, we only need to optimize sensor–actor coordination in each road segment. In the rest of this paper, we formulate the communication cost minimization problem in a road segment and propose a dynamic programming solution and a graph-based algorithm to obtain the optimal solutions. Since a virtual node may not exist in the actual network, the solutions we obtained may be not optimal. However, we will show in Theorem 5 that our approach produces a near-optimal solution, with an asymptotic order of $O(1/m)$, where m is the number of actors in a road segment.

Consider data communication in a road segment between two virtual actors, as shown in Fig. 3. Assume that there are n nodes in the road numbered by $1, 2, \ldots, n$, of which there are m actors, which are denoted by $a_1, a_2, \ldots, a_m$. Notice that $a_1 = 1$ and $a_m = n$ are virtual actors. Sensor i $(1 \leq i \leq n)$ generates sensing data and query data with mean arrival rate λ_i. Actors collect data from nearby sensors and send them to the sink. The sink stores the data, performs data-processing operations, handles user queries, sends back the requested data, broadcasts traffic information, and triggers the appropriate actor to perform some actions such as accident alarm and driver warning.

A. WAS

Our objective is to find a set of working actors to minimize the total network communication cost. We first provide the following formal definition of a WAS.

Definition 1 (WAS): Assume that n sensor and actor nodes are deployed in a road, which are numbered by $\mathcal{S} = \{1, 2, \ldots, n\}$. Among them are m $(m \leq n)$ actors denoted by $\mathcal{A} = \{a_1, a_2, \ldots, a_m\}$ $(1 = a_1 < a_2 < \cdots < a_m = n)$. Suppose that there are K working actors, which are denoted by $w_1, w_2, \ldots, w_K$, where $0 \leq K \leq m$, $1 = w_1 < w_2 < \cdots < w_K = n$, and $w_i \in \mathcal{A}$ $(i = 1, \ldots, K)$. The set $\mathcal{W}_K = \{w_1, w_2, \ldots, w_K\}$ is a subset of $\mathcal{A}$. We call $\mathcal{W}_K$ a WAS of $\mathcal{A}$.

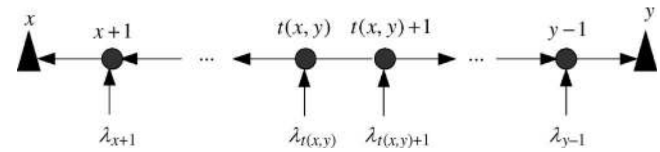

Fig. 4. Data communication between two working actors.

B. Network Communication Cost

It is essential to minimize network communication in a sensor network since sensors and some actors are resource-constrained nodes with limited battery lifetime and communication capabilities, and network communication is a significant component of energy consumption in sensors. Hence, we will use energy cost as the key metric in our analysis.

In RSANs, there are three main components in network communication cost: sensor–sensor, sensor–actor, and actor–sink communication. To simplify the analysis, we assume that the energy cost for unit data transmission in each hop of sensor–sensor and sensor–actor communication is the same, which is denoted by ε. (Note that our model can be extended to the condition where ε is different.) Consequently, the cost for an actor collecting data from a sensor is proportional to the sensor's data generation rate and the number of hops between them. For example, in Fig. 3, the energy cost between actor 1 and sensors 2 and 3 can be calculated by $\lambda_2\varepsilon + 2\lambda_3\varepsilon$. However, energy cost for actor–sink communication is different for each actor since the distance of each actor to the sink is not the same. We use ε_{a_i} to denote the energy cost for actor a_i sending a unit of data to the sink. We also assume that there is a constant energy cost m_{a_i} for keeping an actor in the *working* state, meaning that there is background energy consumption, even in the absence of data transmission. If an actor is a virtual node, we set its ε_{a_i} and m_{a_i} to the minimum value of all working actors.

In our analysis, we use energy consumption to describe communication cost. We should point out that our model could be used in more general situations by applying different cost metrics. For example, if we want to extend the lifetime of the network, we need to reduce the energy consumption of resource-constrained nodes as much as possible. To achieve that, we could assign ε and ε_{a_i} relative cost values: the transmission energy cost to the energy it left. If a node is running out of energy, its relative cost is high; hence, it is less likely to be chosen to perform long-range communications. On the other hand, if an actor is powerful, its ε_{a_i} will be assigned a small value, and as a result, more sensors will transmit data through it. In the rest of this paper, we only use an absolute value of energy consumption as the cost metric to formulate the problem.

Given a WAS $\mathcal{W}_K$, we now formulate its communication cost. We first consider the scenario shown in Fig. 4. In this case, x and y are two working actors, and there is no working actor between them; thus, sensing data in the path are sent to either x or y. Consider a node i $(x < i < y)$; its communication cost to x and y is $|i - x|\lambda_i\varepsilon + \lambda_i\varepsilon_x$ and $|i - y|\lambda_i\varepsilon + \lambda_i\varepsilon_y$ accordingly. To minimize the cost, if i satisfies $|i - x|\lambda_i\varepsilon + \lambda_i\varepsilon_x \leq$

$|i-y|\lambda_i\varepsilon + \lambda_i\varepsilon_y$, i.e., $i \leq (1/2)(x+y-(\varepsilon_x - \varepsilon_y/\varepsilon))$, data from i should be sent to x; otherwise, it should be sent to y. We denote $t(x,y) = \lfloor (1/2)(x+y-(\varepsilon_x - \varepsilon_y/\varepsilon)) \rfloor$ as the *split point* of x and y. As shown in Fig. 4, data from node $x+1,\ldots,t(x,y)$ are sent to actor x, and data from node $t(x,y)+1,\ldots,y-1$ are sent to actor y. If $\varepsilon_x = \varepsilon_y$, $t(x,y)$ is exactly the midpoint of x and y: $t(x,y) = \lfloor (x+y)/2 \rfloor$. We use the function $\phi(x,y)$ to indicate the total communication cost of the nodes between x and y, which can be calculated as

$$\phi(x,y) = \sum_{i=x+1}^{t(x,y)} \lambda_i \cdot (i-x)\varepsilon + \sum_{j=t(x,y)+1}^{y-1} \lambda_j \cdot (y-j)\varepsilon + \sum_{i=x+1}^{t(x,y)} \lambda_i \cdot \varepsilon_x + \sum_{j=t(x,y)+1}^{y-1} \lambda_j \cdot \varepsilon_y. \quad (1)$$

In (1), the first two terms represent the cost for the collection of data from sensor nodes by x and y, and the last two terms represent the cost for sending the data from the actors to the sink. In addition, nodes x and y also act as sensors, which will incur actor–sink communication cost $\lambda_x\varepsilon_x$ and $\lambda_y\varepsilon_y$. Hence, the total communication cost in this scenario is

$$\phi(x,y) + \lambda_x\varepsilon_x + \lambda_y\varepsilon_y + m_x + m_y. \quad (2)$$

Without loss of generality, assume that $\mathcal{W}_K = \{w_1, w_2, \ldots, w_K\}$ $(1 = w_1 < w_2 < \cdots < w_K = n)$ is a WAS. The set of nodes $1, 2, \ldots, n$ can be divided into a number of subsets: $\{w_1,\ldots,w_2\}, \{w_2,\ldots,w_3\},\ldots,\{w_{K-1},\ldots,w_K\}$. The communication cost of each subset can be calculated as in the preceding scenario. Thus, the total communication cost of $\mathcal{W}_K$ is

$$\sum_{i=1}^{K-1} \phi(w_i, w_{i+1}) + \sum_{j\in\mathcal{W}_K} (\lambda_j\varepsilon_j + m_j). \quad (3)$$

C. (a, b)-WorkingSet Problem

We now study the problem of choosing the optimal working actors to minimize the total network communication cost, which we call (a, b)-WorkingSet problem. We provide a formal definition of the problem as follows:

Definition 2 [(a, b)-WorkingSet Problem]: Given a set of sensor and actor nodes $\mathcal{S} = \{a, a+1, \ldots, b\}$ $(a < b)$, a set of actor nodes $\mathcal{A} = \{a_1, a_2, \ldots, a_m\}$ satisfies $a = a_1 < a_2 < \cdots < a_m = b$ $(m \leq b-a)$. Suppose that $\mathcal{W}_K = \{w_1, w_2, \ldots, w_K\}$ is a WAS of $\mathcal{A}$. Define an objective function as

$$C(a,b,\mathcal{W}_K) = \sum_{i=1}^{K-1} \phi(w_i, w_{i+1}) + \sum_{j\in\mathcal{W}_K} (\lambda_j\varepsilon_j + m_j). \quad (4)$$

The (a, b)-WorkingSet problem is given as follows: Find K and $\mathcal{W}_K$ so that $C(a,b,\mathcal{W}_K)$ is minimized.

According to this definition, the sensor–actor coordination problem in road network is simply a $(1,n) - WorkingSet$ problem. Two solutions to the problem are presented in the next section.

III. Solutions

In this section, we present two solutions to the $(a,b) - WorkingSet$ problem: a dynamic programming solution and a graph-based solution.

A. Dynamic Programming Solution

The following theorem shows that a solution to the $(a,b) - WorkingSet$ problem can be obtained by solving subproblems.

Theorem 1: Assume that $\mathcal{S}$ and $\mathcal{A}$ constitute a node set and an actor set, respectively, as defined in Definition 2. Suppose that $\mathcal{W}_K = \{w_1, w_2, \ldots, w_K\}$ is an optimal solution to the $(a,b) - WorkingSet$ problem, and $\mathcal{Q}_\mu = \{q_1, q_2, \ldots, q_\mu\}$ is an optimal solution to the $(a, w_{K-1}) - WorkingSet$ problem; then, $\mathcal{W} = \mathcal{Q}_\mu \cup \{w_K\}$ is also an optimal solution to the $(a,b) - WorkingSet$ problem.

Proof: As $\mathcal{Q}_\mu$ is an optimal solution to the $(a, w_{K-1}) - WorkingSet$ problem, we have

$$C(a, w_{K-1}, \mathcal{Q}_\mu) \leq C\left(a, w_{K-1}, \mathcal{W}_K - \{w_K\}\right). \quad (5)$$

The communication cost of $\mathcal{W}$ is

$$\begin{aligned} C(a,b,\mathcal{W}) &= \left(\sum_{i=1}^{\mu-1} \phi(q_i, q_{i+1}) + \phi(q_\mu, w_K)\right) \\ &\quad + \left(\sum_{j\in\mathcal{Q}_\mu} (\lambda_j\varepsilon_j + m_j) + \lambda_{w_K}\varepsilon_{w_K} + m_{w_K}\right) \\ &= \left(\sum_{i=1}^{\mu-1} \phi(q_i, q_{i+1}) + \sum_{j\in\mathcal{Q}_\mu} (\lambda_j\varepsilon_j + m_j)\right) \\ &\quad + \phi(q_\mu, w_K) + \lambda_{w_K}\varepsilon_{w_K} + m_{w_K} \\ &= C(a, q_\mu, \mathcal{Q}_\mu) + \phi(q_\mu, w_K) + \lambda_{w_K}\varepsilon_{w_K} + m_{w_K}. \end{aligned}$$

On the other hand, the communication cost of $\mathcal{W}_K$ is

$$\begin{aligned} C(a,b,\mathcal{W}_K) &= \sum_{i=1}^{K-1} \phi(w_i, w_{i+1}) + \sum_{j\in\mathcal{W}_K} (\lambda_j\varepsilon_j + m_j) \\ &= \left(\sum_{i=1}^{K-2} \phi(w_i, w_{i+1}) + \sum_{j\in\mathcal{W}_K - \{w_K\}} (\lambda_j\varepsilon_j + m_j)\right) \\ &\quad + \phi(w_{K-1}, w_K) + \lambda_{w_K}\varepsilon_{w_K} + m_{w_K} \\ &= C\left(a, w_{K-1}, \mathcal{W}_K - \{w_K\}\right) + \phi(w_{K-1}, w_K) \\ &\quad + \lambda_{w_K}\varepsilon_{w_K} + m_{w_K}. \end{aligned}$$

Notice that $q_\mu = w_{K-1}$ and that according to (5)

$$\begin{aligned} &C(a,b,\mathcal{W}) - C(a,b,\mathcal{W}_K) \\ &\quad = C(a, w_{K-1}, \mathcal{Q}_\mu) - C\left(a, w_{K-1}, \mathcal{W}_K - \{w_K\}\right) \leq 0. \end{aligned}$$

That is

$$C(a,b,\mathcal{W}) \leq C(a,b,\mathcal{W}_K). \quad (6)$$

On the other hand, since $\mathcal{W}_K$ is an optimal solution to the $(a,b) - WorkingSet$ problem

$$C(a,b,\mathcal{W}_K) \leq C(a,b,\mathcal{W}). \tag{7}$$

Combining (6) and (7), we have

$$C(a,b,\mathcal{W}) = C(a,b,\mathcal{W}_K).$$

Hence, $\mathcal{W}$ is also an optimal solution to the problem. ■

Theorem 1 shows that the problem can be divided into subproblems; thus, dynamic programming can be applied to obtain an optimal solution. By trying W_{K-1} on different actors, the problem can be divided into different subproblems, which form different WASs. The WAS with minimal cost is the optimal solution to the problem. Let $OPT(a,b)$ be the minimum cost obtained in the $(a,b) - WorkingSet$ problem. A dynamic programming solution is given as follows:

$$OPT(a,b) = min\left\{\tau_1, OPT(a,i) + \tau_i\right\} \quad (i = a_2, a_3, \ldots, a_{m-1}) \tag{8}$$

where $\tau_1 = \phi(a,b) + \lambda_a\varepsilon_a + m_a + \lambda_b\varepsilon_b + m_b$ (which is the cost when no working actors are chosen between nodes a and b), and $\tau_i = \phi(i,b) + \lambda_b\varepsilon_b + m_b$.

According to the dynamic programming equation, the minimal cost of the $(1,n) - WorkingSet$ problem can be obtained by calculating $OPT(1,n)$ recursively. At the same time, the optimal WAS $\mathcal{W}_K$ can also be obtained when the process is finished.

Theorem 2: The time complexity of the dynamic programming algorithm in solving a $(1,n) - WorkingSet$ problem is $O(nm^2)$, where n is the number of nodes and m is the number of actors.

Proof: According to (1), the time complexity of $\phi(a,b)$ or $\phi(i,b)$ is $O(n)$. By applying dynamic programming to calculate (8), when we want to get the result of $OPT(a,b)$, the result of $OPT(a,i)$ $(i = a_2, a_3, \ldots, a_{m-1})$ already exists in the system memory. Hence, the value of $OPT(a,b)$ can be obtained by checking $m-1$ values, each of which takes $O(n)$ time in calculating a function ϕ. This process runs in time complexity of $O(mn)$. To calculate $OPT(1,n)$, the result of $OPT(1,a_2), OPT(1,a_3), \ldots, OPT(1,a_{m-1})$ should be sequentially calculated. Thus, the overall time complexity of this problem is $O(n) + O(2n) + \cdots + O((m-1)n)$, which is $O(nm^2)$. ■

B. Graph-Based Solution

In this section, we will introduce a graph-based solution, which is more efficient than the dynamic programming algorithm. First, we construct a *CC-Graph* to represent the communication cost of the $(1,n) - WorkingSet$ problem. Then, we apply a modified Dijkstra's algorithm to find a shortest path in the graph. The shortest path is mapped to a WAS, which is output as an optimal solution to the problem. Finally, we prove the correctness of the graph-based algorithm.

1) Construction of CC-Graph: For a $(1,n) - WorkingSet$ problem, we can use a weighted directed graph to describe its

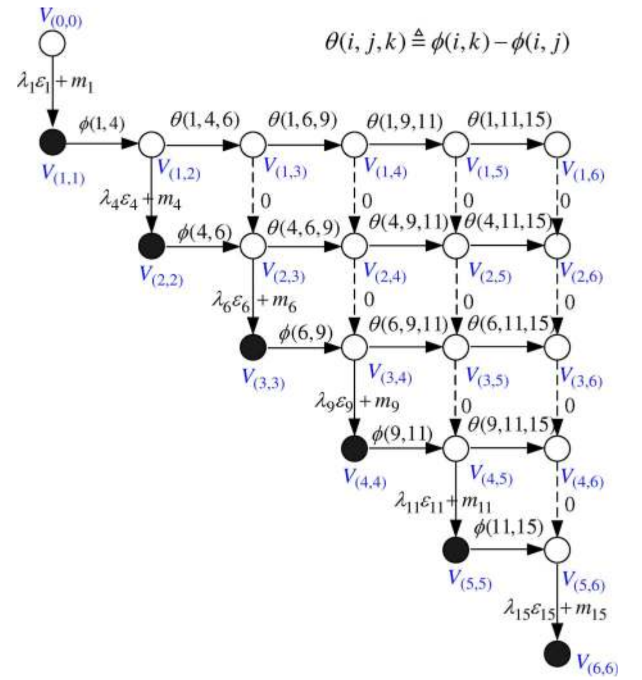

Fig. 5. Example of CC-Graph.

access cost. We call this graph a *CC-Graph*. Fig. 5 gives the corresponding CC-Graph of the road in Fig. 3. The construction of the CC-Graph is described next.

Nodes: As shown in Fig. 5, starting from an *entrance node* $V(0,0)$, the CC-Graph is composed of m rows and m columns, where m is the number of actors. The ith $(1 \leq j \leq m)$ row of the graph has $m+1-i$ nodes, and the jth $(1 \leq j \leq m)$ column has j nodes, forming an inverted triangle. The nodes in row i $(1 \leq i \leq m)$ are denoted as $V_{(i,i)}, V_{(i,i+1)}, \ldots, V_{(i,m)}$ from left to right. We call the last node in each column a *critical node* (marked in black). It can be seen from Fig. 5 that there are m critical nodes in the CC-Graph: $V_{(1,1)}, V_{(2,2)}, \ldots, V_{(m,m)}$. Node $V_{(m,m)}$ is also known as the *exit node*.

Edges: There are horizontal and vertical edges in the CC-Graph. For the ith $(1 \leq i \leq m)$ row, there are horizontal directed edges from $V_{(i,j)}$ to $V_{(i,j+1)}$ $(i \leq j \leq m-1)$, and for the jth $(1 \leq j \leq m)$ column, there are vertical edges from $V_{(i,j)}$ to $V_{(i+1,j)}$ $(1 \leq i < j)$. There is also a vertical edge from the entrance node $V_{(0,0)}$ to node $V_{(1,1)}$. We call the vertical edges from noncritical nodes to critical nodes *solid edges* and the other vertical edges *virtual edges*. We use dashed lines to indicate virtual edges and solid lines to indicate solid edges in the CC-Graph (see Fig. 5).

Weights: We assign a weight for each edge in the CC-Graph. Let $W(V_{(i,j)}, V_{(i,j+1)})$ denote the weight from node $V_{(i,j)}$ to node $V_{(i,j+1)}$. The weights of horizontal edges are calculated by

$$W\left(V_{(i,j)}, V_{(i,j+1)}\right) = \begin{cases} \phi(a_i, a_j), & j = i+1 \\ \phi(a_i, a_{j+1}) - \phi(a_i, a_j), & i+1 < j < m. \end{cases} \tag{9}$$

To reduce computational complexity, we calculate $W(V_{(i,j)}, V_{(i,j+1)})$ as follows:

$$W\left(V_{(i,j)}, V_{(i,j+1)}\right) = \begin{cases} \phi(a_i, a_j), & j = i+1 \\ W_1 + W_2 + W_3, & i+1 < j < m \end{cases} \tag{10}$$

where

$$W_1 = \begin{cases} \sum_{k=t(a_i,a_j)+1}^{t(a_i,a_{j+1})} \left[(2k-a_i-a_j)\lambda_k\varepsilon + (\varepsilon_{a_i}-\varepsilon_{a_j})\lambda_k\right], & t(a_i,a_{j+1}) < a_j \\ \sum_{k=t(a_i,a_j)+1}^{a_j-1} \left[(2k-a_i-a_j)\lambda_k\varepsilon + (\varepsilon_{a_i}-\varepsilon_{a_j})\lambda_k\right], & a_j \le t(a_i,a_{j+1}) \end{cases}$$

$$W_2 = \begin{cases} \left[(a_{j+1}-a_j)\varepsilon + (\varepsilon_{a_{j+1}}-\varepsilon_{a_j})\right] \times \sum_{k=t(a_i,a_{j+1})+1}^{a_j-1} \lambda_k, & t(a_i,a_{j+1}) < a_j \\ \sum_{k=a_j}^{t(a_i,a_{j+1})} \left[(k-a_i)\lambda_i\varepsilon + \lambda_i\varepsilon_{a_i}\right], & a_j \le t(a_i,a_{j+1}) \end{cases}$$

$$W_3 = \begin{cases} \sum_{k=a_j}^{a_{j+1}-1} \left[(a_{j+1}-k)\lambda_k\varepsilon + \lambda_k\varepsilon_{a_{j+1}}\right], & t(a_i,a_{j+1}) < a_j \\ \sum_{k=t(a_i,a_{j+1})+1}^{a_{j+1}-1} \left[(a_{j+1}-k)\lambda_k\varepsilon + \lambda_k\varepsilon_{a_{j+1}}\right], & a_j \le t(a_i,a_{j+1}). \end{cases}$$

The equality of (9) and (10) can be proven by using the definition of ϕ (1). The detailed derivation is shown in the Appendix.

For vertical edges, the weight of a virtual edge is assigned 0, and the weight of a solid edge is assigned the communication cost of the corresponding actor

$$W\left(V_{(i,j)}, V_{(i+1,j)}\right) = \begin{cases} \lambda_{a_j}\varepsilon_{a_j} + m_{a_j}, & i = j-1 \\ 0, & \text{otherwise} \end{cases}$$
$$(1 \le i < j, 2 \le j \le m). \tag{11}$$

In addition, the weight of edge $V(0,0) \to V(1,1)$ is assigned by

$$W\left(V_{(0,0)}, V_{(1,1)}\right) = \lambda_1\varepsilon_1 + m_1.$$

CC-Path and graph traveling rule: We define a *communication cost path (CC-Path)* as a path from the entrance node $V_{(0,0)}$ to the exit node $V_{(m,m)}$ in the CC-Graph. For example, in Fig. 5, a CC-Path is the path from $V_{(0,0)}$ to $V_{(1,1)}$, traversing to $V_{(1,3)}$, then down to $V_{(3,3)}$, then to $V_{(3,5)}$, then down to $V_{(5,5)}$, then to $V_{(5,6)}$, finally reaching $V_{(6,6)}$.

Traveling on a CC-Graph is like walking through a CC-Path from the entrance node to the exit node. We make the following graph traversal rule: each time a virtual edge is traversed, the traversal vertically continues until a solid edge is reached. In other words, both horizontal and vertical traversals are possible. Direction can be changed any time in horizontal traversals, whereas in vertical traversals, a critical node must be reached before direction can be changed.

By applying the graph traveling rule, a CC-Path can be denoted by a sequence of critical nodes that it passes through. For example, the aforementioned CC-Path passes through four critical nodes, which can be denoted by $(V_{(1,1)}, V_{(3,3)}, V_{(5,5)}, V_{(6,6)})$. Obviously, each CC-Path must contain the entrance node and exit node. On the other hand, each critical node sequence containing $V_{(1,1)}$ and $V_{(m,m)}$ denotes a unique CC-Path, i.e., the sequence $(V_{(1,1)}, V_{(m-1,m-1)}, V_{(m,m)})$ denotes the CC-Path: $V_{(0,0)} \to V_{(1,1)} \to \cdots \to V_{(1,m-1)} \cdots \to V_{(m-1,m-1)} \to V_{(m-1,m)} \to V_{(m,m)}$.

2) Graph-Based Algorithm: The graph-based algorithm is very simple. The basic idea is to construct a CC-Graph and find the shortest CC-Path in the graph. The intuition is that the sum of weights in a CC-Graph represents the communication cost of a WAS. Thus, the shortest path represents the lowest cost WAS. It is easy to modify the Dijkstra's algorithm to follow our graph traveling rule in finding a shortest CC-Graph. The detailed steps are described as follows.

Step 1) Construct a CC-Graph, and assign weights to its edges using the method introduced in Section III-B1.

Step 2) Apply a modified Dijkstra's algorithm on the CC-Graph with its weight assignment to get a shortest path.

Step 3) Assuming that the shortest cost path finding in step 2 is represented by a sequence of critical nodes $(V_{(x_1,x_1)}, V_{(x_2,x_2)}, \ldots, V_{(x_k,x_k)})$ $(1 = x_1 < x_2 < \cdots < x_k = m)$, map the sequence to a set of working actor nodes $\mathcal{W} = \{x_1, x_2, \ldots, x_k\}$. Output $\mathcal{W}$ as the optimal WAS.

The correctness of the graph-based algorithm will be proven in the next section.

3) Correctness of the Graph-Based Algorithm: We will now prove the correctness of the graph-based algorithm. Several lemmas are needed to get the desired result represented by Theorem 3.

Lemma 1: Each CC-Path in the graph corresponds to a WAS of actor nodes; each WAS of the actor node corresponds to a CC-Path in the graph.

Proof: We will show that a CC-path can be mapped to a WAS of actor nodes. It has been shown in Section III-B1 that a CC-Path can be denoted by a sequence of critical nodes. Assume that $(V_{(x_1,x_1)}, V_{(x_2,x_2)}, \ldots, V_{(x_k,x_k)})$ $(1 = x_1 < x_2 < \cdots < x_k = m)$ is such a sequence. We now map it to a set of working actors: $\mathcal{W} = \{a_{x_1}, a_{x_2}, \ldots, a_{x_k}\}$. $\mathcal{W}$ is a subset of actors $\{a_1, a_2, \ldots, a_m\}$. According to Definition 1, $\mathcal{W}$ is a WAS of actor nodes.

Now, we show that a WAS can be mapped to a CC-Path. Assume that $\mathcal{W} = \{a_{x_1}, a_{x_2}, \ldots, a_{x_k}\}$ $(1 = x_1 < x_2 < \cdots < x_k = m)$ is a WAS of actor nodes. We can map it to a critical node sequence $(V_{(x_1,x_1)}, V_{(x_2,x_2)}, \ldots, V_{(x_k,x_k)})$. According to the definition of CC-Path, this sequence corresponds to a CC-Path in the graph. The lemma is proven. ■

Lemma 2: The sum of the weights in a CC-Path is equal to the communication cost of the corresponding WAS.

Proof: According to Lemma 1, a CC-Path can be denoted by a critical node sequence $(V_{(x_1,x_1)}, V_{(x_2,x_2)}, \ldots,$

$V_{(x_k,x_k)})$ $(1 = x_1 < x_2 < \cdots < x_k = m)$, and its corresponding WAS is $W = \{x_1, x_2, \ldots, x_k\}$. The CC-Path contains $k-1$ horizontal subpaths and k vertical subpaths.

The horizontal subpaths and their weights are given as follows:

$\boxed{1}$ $V_{(x_1,x_1)} \rightarrow \cdots \rightarrow V_{(x_1,x_2)}$
whose sum of weights is

$$\sum_{j=x_1}^{x_2-1} W\left(V_{(x_1,j)}, V_{(x_1,j+1)}\right) = \phi(x_1, x_1+1) + \sum_{j=x_1+1}^{x_2-1} \left(\phi(x_1, j+1) - \phi(x_1, j)\right) = \phi(x_1, x_2);$$

$\boxed{2}$ $V_{(x_2,x_2)} \rightarrow \cdots \rightarrow V_{(x_2,x_3)}$
whose sum of weights is

$$\sum_{j=x_2}^{x_3-1} W\left(V_{(x_2,j)}, V_{(x_2,j+1)}\right) = \phi(x_2, x_2+1) + \sum_{j=x_2+1}^{x_3-1} \left(\phi(x_2, j+1) - \phi(x_2, j)\right) = \phi(x_2, x_3)$$

$\vdots$

$\boxed{k-1}$ $V_{(x_{k-1},x_{k-1})} \rightarrow \cdots \rightarrow V_{(x_{k-1},x_k)}$
whose sum of weights is

$$\sum_{j=x_{k-1}}^{x_k-1} W\left(V_{(x_{k-1},j)}, V_{(x_{k-1},j+1)}\right) = \phi(x_{k-1}, x_k).$$

The k vertical subpaths and their weights are given as follows:

$\boxed{1}$ $V_{(0,0)} \rightarrow V_{(x_1,x_1)}$
whose sum of weights is $\lambda_{x_1}\varepsilon_{x_1} + m_{x_1}$;

$\boxed{2}$ $V_{(x_1,x_2)} \rightarrow \cdots \rightarrow V_{(x_2,x_2)}$
whose sum of weights is $\lambda_{x_2}\varepsilon_{x_2} + m_{x_2}$;

$\vdots$

$\boxed{k}$ $V_{(x_{k-1},x_k)} \rightarrow \cdots \rightarrow V_{(x_k,x_k)}$
whose sum of weights is $\lambda_{x_k}\varepsilon_{x_k} + m_{x_k}$.

Thus, the sum of the weights of the CC-Path is

$$\sum_{i=1}^{k-1} \phi(x_i, x_{i+1}) + \sum_{j\in W} (\lambda_j \varepsilon_j + m_j). \tag{12}$$

Compared with (3), the sum of the weights in a CC-Path is equal to the communication cost of the W. The lemma is proven. ∎

Theorem 3: The graph-based algorithm gives an optimal solution to the $(1, n) - WorkingSet$ problem.

Proof: According to Lemma 1, there exists a one-to-one mapping between the CC-Paths and the possible WAS of m actor nodes. Lemma 2 shows that the sum of weights of a CC-Path is equal to the communication cost of the corresponding WAS. By applying the modified Dijkstra's algorithm, a shortest path can be found. The sum of weights of the shortest path corresponds to the WAS with minimal communication cost, which is the optimal solution to the $(1, n) - WorkingSet$ problem. ∎

Theorem 4: The time complexity of the graph-based algorithm in solving a $(1, n) - WorkingSet$ problem is $O(mn + m^2 lgm)$.

Proof: The graph-based algorithm contains three steps. We demonstrate the running time of each step as follows: For step 1, a CC-Graph has $O(m^2)$ nodes and $O(m^2)$ edges. According to (10), the running time of calculating $W(V_{(i,j)}, V_{(i,j+1)})$ is $O(t(a_i, a_{j+1}) - t(a_i, a_j)) + O(a_{j+1} - a_j)$. Thus, the running time for calculating the weights of edges in each row is $O(\sum_{j=i+1}^{m} W(V_{(i,j)}, V_{(i,j+1)})) = O(t(a_i, a_m)) + O(a_m) = O(n)$. Thus, the time complexity of CC-Graph construction in step 1 is $O(mn)$. According to [18], the running time of Dijkstra's algorithm in a graph with V nodes and E edges is $O(VlgV + E)$ by using a Fibonacci heap implementation. In our case, the running time of applying the modified Dijkstra's algorithm in an CC-Graph (step 2) is $O(m^2 lg(m^2) + m^2) = O(m^2 lgm)$. The running time of step 3 is clearly $O(m)$. Thus, the total running time of the graph-based algorithm is $O(mn + m^2 lgm)$. ∎

Compared with Theorem 2, it can be seen that the graph-based algorithm is more efficient than the dynamic programming algorithm.

C. Discussions

As we have mentioned, the dynamic programming algorithm and the graph-based algorithm produce an optimal solution based on the assumption of virtual working actors. In real RSANs, working actors may be not deployed in each intersection; thus, the solution may be not optimal in the real case. The following theorem shows that the proposed algorithms produce a near-optimal solution when there are no working actors in the intersections.

Theorem 5: Given n sensors and actors deployed in a road segment, with m actors among them, assume that C is the minimum communication cost obtained by the dynamic programming algorithm or graph-based algorithm, and C' is the actual communication cost. Let $\Delta C = C' - C$, and consider the expectation of $\Delta C/C$. We have

$$E\left(\frac{\Delta C}{C}\right) = O\left(\frac{1}{m}\right). \tag{13}$$

Proof: Assume that $S = \{1, 2, \ldots, n\}$ is the set of sensors and actors. We now consider the first node to be a virtual node. Due to symmetry, the situation is similar when the last node is a virtual node. If there is no actual working actor deployed in the first node, the solution obtained by the dynamic programming algorithm and graph-based algorithm is not optimal since the traffic to the first node needs to be redirected to a nearby

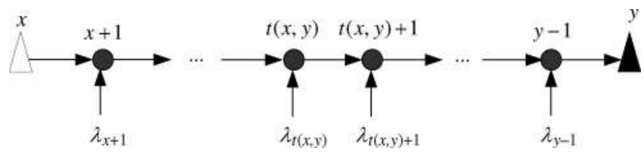

Fig. 6. Communication cost when the first node is not a working actor.

working actor. Assume that y $(1 < y < n)$ is the first working actor from the first node. As shown in Fig. 6, the traffic in the path from x to $t(x, y)$ ($x = 1$ *in this case*) is directed to y, but the traffic of the rest of the nodes remains unchanged.

According to (1), if the first node is a working actor, the communication cost is

$$\phi(1,y) = \sum_{i=1}^{t(1,y)} \lambda_i \cdot (i-1)\varepsilon + \sum_{j=t(1,y)+1}^{y-1} \lambda_j \cdot (y-j)\varepsilon + \sum_{i=1}^{t(1,y)} \lambda_i \cdot \varepsilon_1 + \sum_{j=t(1,y)+1}^{y-1} \lambda_j \cdot \varepsilon_y.$$

If the first node is not a working actor, as shown in Fig. 6, the communication cost becomes

$$\phi'(1,y) = \sum_{i=1}^{t(1,y)} \lambda_i \cdot (y-i)\varepsilon + \sum_{j=t(1,y)+1}^{y-1} \lambda_j \cdot (y-j)\varepsilon + \sum_{i=1}^{y} \lambda_i \cdot \varepsilon_y.$$

Thus

$$\Delta C = \phi'(1,y) - \phi(1,y) = \sum_{i=1}^{t(1,y)} \lambda_i (y+1-2i) + \sum_{i=1}^{t(1,y)} \lambda_i (\varepsilon_y - \varepsilon_1).$$

Let $\lambda_{\max} = \max\{\lambda_i\}$ and $\lambda_{\min} = \min\{\lambda_i\}$, $(i = 1, 2, \ldots, n)$. We have

$$\Delta C \leq y^2 \lambda_{\max} + y\lambda_{\max}(\varepsilon_y - \varepsilon_1) = O(y^2).$$

Assume that m actors are randomly deployed in the n nodes, i.e., each node has a probability of m/n being an actor. Thus, the probability of the first working actor being the ith node is

$$Pr\{y = i\} = \left(1 - \frac{m}{n}\right)^{i-1} \cdot \frac{m}{n}.$$

Thus, the expectation of y^2 is

$$E(y^2) = \sum_{i=1}^{n} i^2 \left(1 - \frac{m}{n}\right)^{i-1} \frac{m}{n} \approx 1 + 3\frac{n-m}{m} + 2\left(\frac{n-m}{m}\right)^2.$$

On the other hand, by (1)

$$\begin{aligned}\phi(x,y) &\geq \sum_{i=x+1}^{t(x,y)} \lambda_i \cdot (i-x)\varepsilon + \sum_{j=t(x,y)+1}^{y-1} \lambda_j \cdot (y-j)\varepsilon \\ &\geq \lambda_{\min}\varepsilon \left(\sum_{i=x+1}^{t(x,y)} (i-x) + \sum_{j=t(x,y)+1}^{y-1} (y-j) \right) \\ &\geq \lambda_{\min}\varepsilon \left(1 + \frac{y-x}{2}\right)(y-x).\end{aligned}$$

Fig. 7. Simulation area.

According to (3), the optimal cost C satisfies

$$\begin{aligned}C &\geq \sum_{i=1}^{m} \phi(w_i, w_{i+1}) \geq m\lambda_{\min}\varepsilon \left(1 + \frac{n}{2m}\right)\frac{n}{m} \\ &= \lambda_{\min}\varepsilon \left(1 + \frac{n}{2m}\right) n.\end{aligned}$$

Thus

$$\begin{aligned}E\left(\frac{\Delta C}{C}\right) &= O\left(\frac{E(y^2)}{C}\right) = O\left(\frac{1 + 3\frac{n-m}{m} + 2\left(\frac{n-m}{m}\right)^2}{\lambda_{\min}\varepsilon\left(1 + \frac{n}{2m}\right) n}\right) \\ &= O\left(\frac{1}{m}\right).\end{aligned}$$

The theorem is proven. ■

Theorem 5 shows that, without deploying virtual nodes, the proposed algorithms will give a near-optimal solution, which approaches the optimal solution with asymptotic order $O(1/m)$. If the number of actors m in a road segment is large enough, our solution will be very close to the optimal solution.

IV. Performance Evaluation

In this section, we conduct simulation experiments to evaluate the performance of the proposed algorithms. Our simulation scenario is based on the application of road condition surveillance in a city block of Beijing, China. Experiment results show the efficiency of the proposed strategies.

A. Simulation Environment

We simulate the application of road surveillance for the Beijing Olympic Stadium. Fig. 7 shows a rectangular area (2980 m × 2030 m), where the Beijing Olympic Stadium is located. Sensors and actors are randomly deployed in the three main roads from east to west and the two main roads from north to south (yellow lines in the figure) around the Olympic Stadium to monitor road conditions. When sensors detect events, the sensing data are sent to the nearby actors and

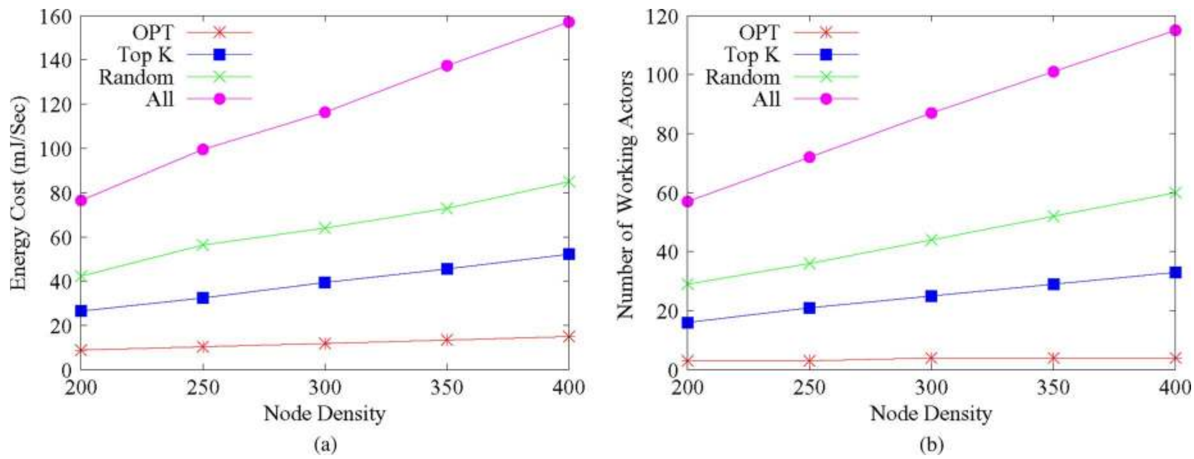

Fig. 8. Performance under various node density ($\overline{\lambda} = 0.01$). (a) Energy cost. (b) Number of working actors.

then onward to the sink, which is located inside the Olympic Sports Center, as shown in the figure.

We list a number of tunable parameters used in the experiments as follows: We use ρ to indicate *node density*, which is calculated by the average number of nodes per kilometer in the WSAN. We vary ρ from 200 to 400 to check the scalability of the algorithm. ξ represents the *actor density* (the average number of actors per kilometer), varying from 30 to 150 in the experiments.

We consider three traffic conditions: 1) busy traffic (representing rush hours), where each sensor generates hundreds of sensing data per second; 2) normal, where each sensor generates one or two units of data per second; 3) idle (representing late evening or early morning), where only a few sensing data are generated in a long interval. We use $\overline{\lambda}$ to denote the average data generation rate of a sensor. $\overline{\lambda} = 100, 1, 0.01$ corresponds to the preceding three conditions, respectively. Each sensing data unit is assumed to be a single byte. The transmission range of sensors is set to 10 m.

We employ an energy model similar to [3] and [10]. The energy consumption per bit is calculated as $E = 2E_{\mathrm{elec}} + \beta d^{\alpha}$, where α is the exponent of the path loss $(2 \leq \alpha \leq 5)$, β is a constant $[J/(bit \cdot m^{\alpha})]$, and E_{elec} is the energy needed by the transceiver circuitry to transmit or receive one bit [in joules per bit]. The communication energy parameters are set as $E_{\mathrm{elec}} =$ 50 nJ/bit, $\beta = 10$ pJ/bit/m^2, and $\alpha = 2$.

As both the dynamic programming algorithm and graph-based algorithm give near-optimal solutions to the problem, we only implement the graph-based algorithm in our simulation for its complexity is lower. The graph-based algorithm is referred to the OPT algorithm in the rest of this paper. We compare our algorithm with several other heuristic solutions: 1) *Random*: Select working actors at random. 2) *Top* K: Select the K largest aggregate data transmission rate actors as the working set. The top 30% actors are chosen in our simulation. 3) *ALL*: All actors are assumed to be in the working state.

B. Simulation Results

Fig. 8 shows performance under the traffic condition $\overline{\lambda} =$ 0.01. Fig. 8(a) compares system energy consumption as a function of node density ρ. When ρ varies from 200 to 400, the energy cost of Random, Top K, and ALL increases, but the cost of OPT remains constant. As can be seen, OPT has the lowest energy cost and is invariant to node density. As expected, ALL has the highest energy cost, which is about dozens of times higher than that of the OPT strategy. Fig. 8(b) shows the average number of working actors selected by the four different algorithms on each road. The OPT strategy shows a near constant number of working actors under all conditions. There are many more working actors in the other algorithms, which linearly increase with the node density.

The reason OPT consumes much less energy than the other algorithms can be explained as follows: For the condition $\overline{\lambda} = 0.01$, there is only a few data items to be transferred in the network. Actors are idle most of the time. The OPT strategy turns off as many actors as possible to save energy. As shown in Fig. 8(b), OPT selects about four working actors and is insensitive to node density. Thus, the energy cost remains constant. The Random, Top K, and ALL strategies choose more working actors while the node density increases. As a result, more energy is consumed, even if there is no data transmission.

Fig. 9 shows the performance when the node density is fixed to 300 and the actor density varies from 30 to 150. It is shown that the energy cost of all algorithms, with the exception of OPT, increases with the node density. Again, OPT demonstrates a constant number of working actors and the lowest energy cost.

Figs. 10 and 11 compare performance for $\overline{\lambda} = 1$. In this case, the energy cost and working actor number of all algorithms increase when the node density increases. Still, OPT has the lowest energy cost. The performances of Random and Top K are similar: they both consume more energy than OPT. ALL has the highest energy cost, which is about 10%–20% higher than OPT. Fig. 11 also shows a favorable property of the OPT algorithm: the number of working actors is insensitive to the actor density ξ. For the OPT algorithm, the number of working actors remains constant when ξ varies from 30 to 150. Its corresponding energy cost also remains the same most of the time. Other algorithms use more working actors and consume more energy than OPT algorithm. In this figure, the energy cost of ALL varies from 675 to 850 mJ when the actor density increased. However, the cost of OPT remains below 650 mJ,

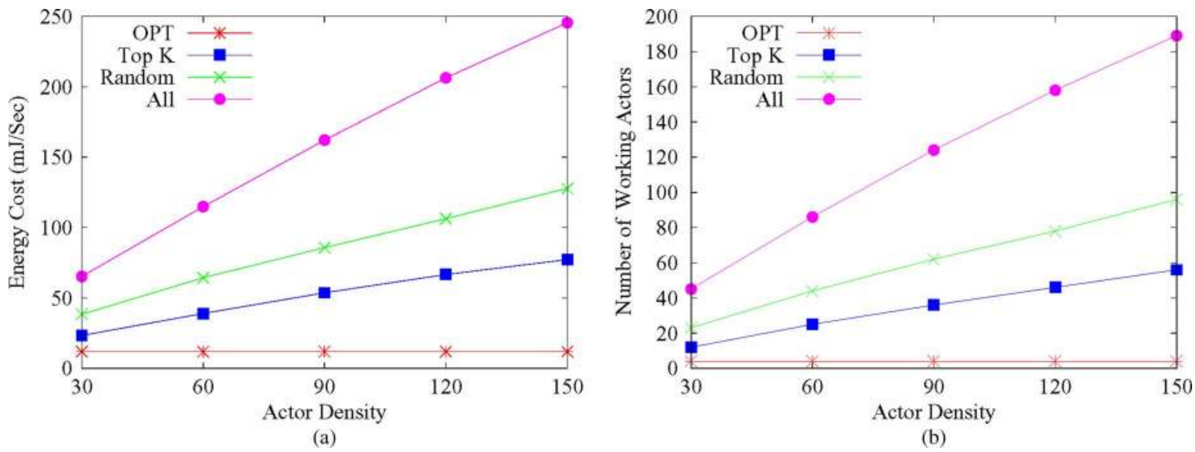

Fig. 9. Performance under various actor density ($\rho = 300, \overline{\lambda} = 0.01$). (a) Energy cost. (b) Number of working actors.

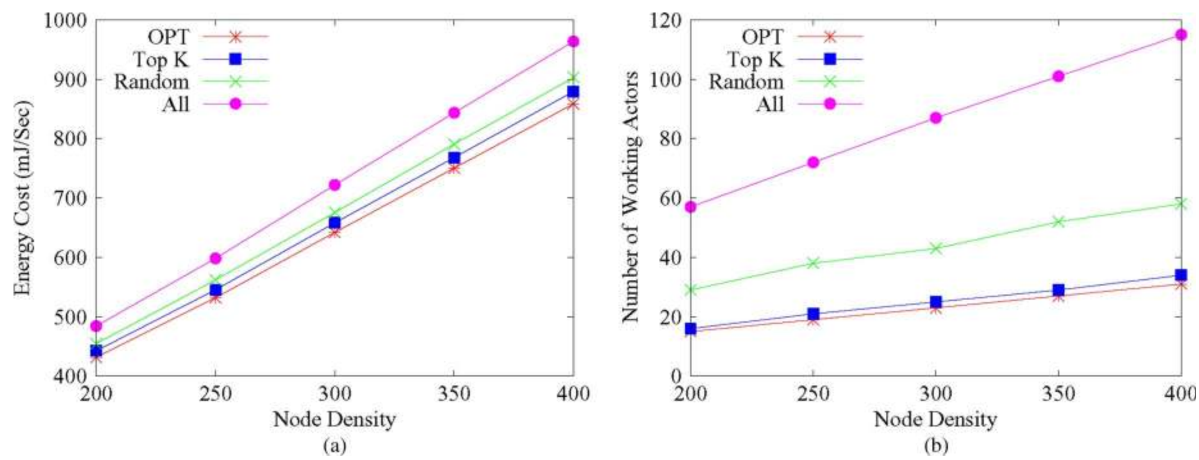

Fig. 10. Performance under various node density ($\overline{\lambda} = 1$). (a) Energy cost. (b) Number of working actors.

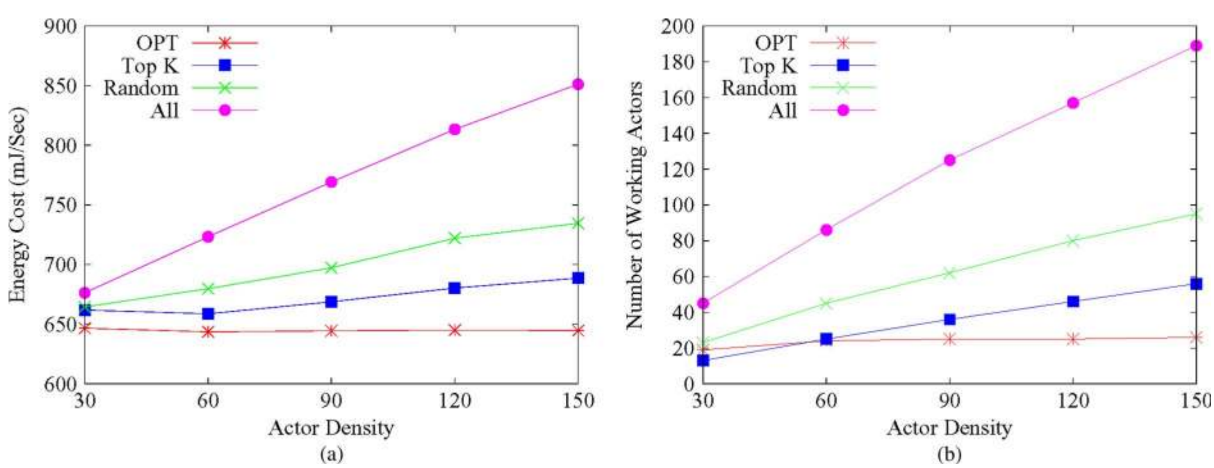

Fig. 11. Performance under various actor density ($\rho = 300, \overline{\lambda} = 1$). (a) Energy cost. (b) Number of working actors.

which represents an energy savings of about 30%, compared with ALL.

Fig. 12 compares performance for the different schemes when $\overline{\lambda} = 100$. As illustrated in the figure, even though the number of working actors selected by these algorithms is quite different, the energy costs of the four strategies are roughly the same. The reason is given as follows: Unlike the other two conditions, data traffic is quite busy in this case; thus, data transmission dominates energy consumption. Fig. 13(a) shows the percentage of energy consumed in long-range actor–sink communication. As can be seen, more than 90% of the energy is consumed in actor–sink communication. When there is a sufficient number of working actors, adding more working actors only affects data communication between sensors, which cannot reduce the cost of actor–sink communication. Thus, energy cost is insensitive to actor-selecting algorithms when

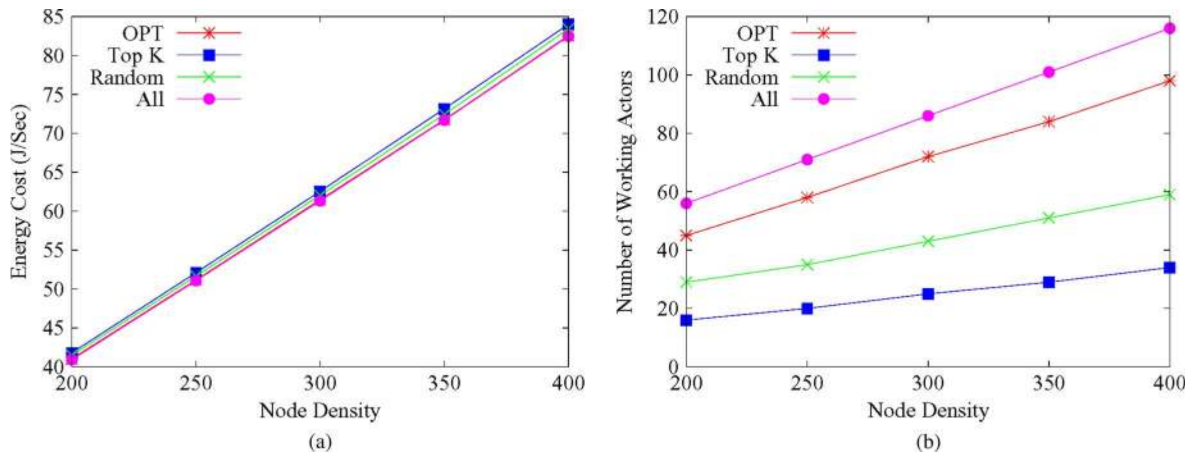

Fig. 12. Performance under various node density ($\overline{\lambda} = 100$). (a) Energy cost. (b) Number of working actors.

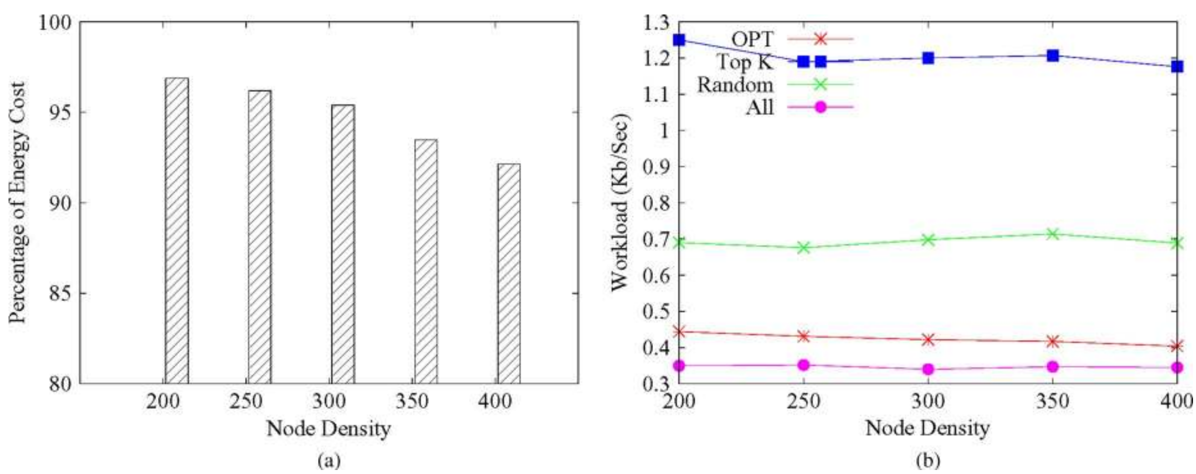

Fig. 13. Energy and workload under various node density ($\overline{\lambda} = 100$). (a) Proportion of actor–sink energy cost. (b) Average workload.

traffic is busy. However, the number of actors does affect the average workload of each node. Fig. 13(b) compares the average data transmission rate of the algorithms. It shows that OPT has the lowest average workload, as all actors are working. The number of working actors in OPT is very close to ALL [see Fig. 12(b)]; thus, it also has a reasonable workload. TOP K has the heaviest workload, which is more than twice that of the OPT algorithm.

In conclusion, according to the simulation, the advantages of our algorithm can be summarized as follows: When the system is in idle condition, the OPT algorithm turns off as many actors as possible to save energy. For normal road surveillance, it yields the lowest energy cost, and its performance is insensitive to actor density. In a busy traffic environment, it keeps the workload of working actors at a low level.

V. Related Work

Unlike traditional wireless sensor networks, which are composed of homogeneous nodes and single-sink performing centralized operations [1], WSANs enable heterogeneous nodes to cooperatively perform distributed sensing and active tasks [2]–[4]. Equipped with more energy and computing resources, the actor nodes can perform more complex functions such as data collection, coordination, and responses to events. However, the heterogeneity of WSANs also introduces many research challenges including node deployment, real-time requirement, coordination, mobility, etc. [2]–[4], [19], [20].

The use of WSANs for vehicular tracking has been proposed in a number of recent projects [5]–[8]. Karpinski *et al.* proposed a software architecture for sensor networks in smart roads [6]. Jeong *et al.* presented an autonomous passive localization scheme for road sensor networks [5]. However, little attention has been paid to effective coordination in sensors and actors in RSANs.

Some recent papers have considered the issue of real-time communication and coordination in sensor networks [2], [3], [21]. The SPEED protocol [21] provides real-time communication services and is designed to be a stateless localized algorithm with low control overhead. Melodia *et al.* proposed a sensor–actor coordination model based on an event-driven partitioning paradigm to achieve global network objectives such as real-time constraints and minimum energy consumption [3]. Wu *et al.* studied sensor–actor connectivity strategies to put as many sensors as possible to sleep for energy-saving purposes [2].

Energy-efficient routing and data-collecting protocols have been extensively studied, and a wide range of special network structures, such as shortest path tree, minimum spanning tree, minimum Steiner tree, clustering, grid, and chain, are utilized to achieve the goal of reducing energy consumption [3], [10]–[17], [22]–[26]. Sleep management has been shown to be an effective method for saving energy [27]–[29]. The protocols proposed in [30] and [31] can improve network performance by maintaining a backbone composed of a small number of active nodes while scheduling other nodes to operate in low duty cycles to conserve energy. However, as we mentioned before, these strategies cannot be directly used in RSANs. To our knowledge, the sensor–actor coordination issues with the explicit objective of minimizing network communication cost in road networks has not been extensively studied in the past.

VI. Conclusion

In this paper, we have addressed the question of choosing a set of working actors for coordinating data transmission in WSANs to minimize communication cost. We have first presented a theoretical model to analyze the communication cost of data transmission in RSANs. Based on the assumption of virtual nodes, we have formulated the sensor–actor coordinating communication problem as an optimization problem and proven that it can be divided into subproblems, thus allowing optimal solutions to be obtained by using dynamic programming. The time complexity of the dynamic programming algorithm has been proven to be $O(nm^2)$. We have also proposed a lower complexity graph-based algorithm, which uses a CC-Graph to depict the cost of data communication and a modified Dijkstra's algorithm to find an optimal solution to the problem. The running time of the graph-based algorithm is $O(mn + m^2 lgm)$, which is more efficient than the dynamic programming algorithm and more applicable in large-scale distributed systems. We have further proven that, without the deployment of actors in virtual nodes, the proposed strategies produce a near-optimal solution, which approaches the optimal solution with asymptotic order $O(1/m)$. Performance of the proposed algorithms has been evaluated by simulation, which confirms the efficiency of the proposed algorithms.

The strategies discussed in this paper are local optimization based on road segments divided by virtual nodes. In the future, we will address the following issues to improve it. We intend to consider how to minimize communication cost in the entire network, instead of individual segments, which is a substantially more complicated optimization problem. We will also investigate the feasibility of exploiting the mobility pattern of the vehicles to carry the sensing data to the nearby actors as a means to decrease the cost of sensor–actor communications.

Appendix
Proof of the Equality of (9) and (10)

The equality of (9) and (10) can be verified by showing that $\phi(a_i, a_{j+1}) - \phi(a_i, a_j) = W_1 + W_2 + W_3$ when $i + 1 < j < m$.

There are two possible relative positions of node a_j and the split point of a_i and a_{j+1}: $t(a_i, a_{j+1}) < a_j$, and $a_j \leq t(a_i, a_{j+1})$.

If $t(a_i, a_{j+1}) < a_j$, i.e., node a_j is in the right part of the midpoint $t(a_i, a_{j+1})$, according to (1), we have

$$\begin{aligned}
\phi(a_i, a_j) =& \sum_{k=a_i+1}^{t(a_i,a_j)} \lambda_k \cdot (k-a_i)\varepsilon + \sum_{k=t(a_i,a_j)+1}^{a_j-1} \lambda_k \cdot (a_j-k)\varepsilon \\
&+ \sum_{k=a_i+1}^{t(a_i,a_j)} \lambda_k \cdot \varepsilon_{a_i} + \sum_{k=t(a_i,a_j)+1}^{a_j-1} \lambda_k \cdot \varepsilon_{a_j} \\
=& \sum_{k=a_i+1}^{t(a_i,a_j)} [\lambda_k \cdot (k-a_i)\varepsilon + \lambda_k \cdot \varepsilon_{a_i}] \\
&+ \sum_{k=t(a_i,a_j)+1}^{t(a_i,a_{j+1})} \left[\lambda_k \cdot (a_j-k)\varepsilon + \lambda_k \cdot \varepsilon_{a_j}\right] \\
&+ \sum_{k=t(a_i,a_{j+1})+1}^{a_j-1} \left[\lambda_k \cdot (a_j-k)\varepsilon + \lambda_k \cdot \varepsilon_{a_j}\right]
\end{aligned}$$

$$\begin{aligned}
\phi(a_i, a_{j+1}) =& \sum_{k=a_i+1}^{t(a_i,a_{j+1})} \lambda_k \cdot (k-a_i)\varepsilon \\
&+ \sum_{k=t(a_i,a_{j+1})+1}^{a_{j+1}-1} \lambda_k \cdot (a_{j+1}-k)\varepsilon \\
&+ \sum_{k=a_i+1}^{t(a_i,a_{j+1})} \lambda_k \cdot \varepsilon_{a_i} + \sum_{k=t(a_i,a_{j+1})+1}^{a_{j+1}-1} \lambda_k \cdot \varepsilon_{a_{j+1}} \\
=& \sum_{k=a_i+1}^{t(a_i,a_j)} [\lambda_k \cdot (k-a_i)\varepsilon + \lambda_k \cdot \varepsilon_{a_i}] \\
&+ \sum_{k=t(a_i,a_j)+1}^{t(a_i,a_{j+1})} [\lambda_k \cdot (k-a_i)\varepsilon + \lambda_k \cdot \varepsilon_{a_i}] \\
&+ \sum_{k=t(a_i,a_{j+1})+1}^{a_j-1} \left[\lambda_k \cdot (a_{j+1}-k)\varepsilon + \lambda_k \cdot \varepsilon_{a_{j+1}}\right] \\
&+ \sum_{k=a_j}^{a_{j+1}-1} \left[\lambda_k \cdot (a_{j+1}-k)\varepsilon + \lambda_k \cdot \varepsilon_{a_{j+1}}\right].
\end{aligned}$$

Thus

$$\begin{aligned}
&\phi(a_i, a_{j+1}) - \phi(a_i, a_j) \\
&\quad= \sum_{k=t(a_i,a_j)+1}^{t(a_i,a_{j+1})} \left[(2k - a_i - a_j)\lambda_k\varepsilon + (\varepsilon_{a_i} - \varepsilon_{a_j})\lambda_k\right] \\
&\qquad+ \left[(a_{j+1} - a_j)\varepsilon + (\varepsilon_{a_{j+1}} - \varepsilon_{a_j})\right] \sum_{k=t(a_i,a_{j+1})+1}^{a_j-1} \lambda_k \\
&\qquad+ \sum_{k=a_j}^{a_{j+1}-1} \left[(a_{j+1} - k)\lambda_k\varepsilon + \lambda_k\varepsilon_{a_{j+1}}\right].
\end{aligned}$$

Compared with the expression of W_1, W_2, and W_3, when $t(a_i, a_{j+1}) < a_j$, it is easy to see that

$$\phi(a_i, a_{j+1}) - \phi(a_i, a_j) = W_1 + W_2 + W_3.$$

If $a_j \leq t(a_i, a_{j+1})$, i.e., node a_j is in the left of the midpoint $t(a_i, a_{j+1})$, we can obtain the same conclusion similarly. Thus, the equality of (9) and (10) is proven.

REFERENCES

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: A survey," *Comput. Netw.*, vol. 38, no. 4, pp. 393–422, Mar. 2002.

[2] J. Wu, S. Yang, and M. Cardei, "On maintaining sensor–actor connectivity in wireless sensor and actor networks," in *Proc. IEEE INFOCOM*, Apr. 2008, pp. 286–290.

[3] T. Melodia, D. Pompili, V. Gungor, and I. Akyildiz, "Communication and coordination in wireless sensor and actor networks," *IEEE Trans. Mobile Comput.*, vol. 6, no. 10, pp. 1116–1129, Oct. 2007.

[4] I. F. Akyildiz and I. H. Kasimoglu, "Wireless sensor and actor networks: Research challenges," *Ad Hoc Netw.*, vol. 2, no. 4, pp. 351–367, Oct. 2004.

[5] J. Jeong, S. Guo, T. He, and D. Du, "APL: Autonomous passive localization for wireless sensors deployed in road networks," in *Proc. IEEE INFOCOM*, Apr. 2008, pp. 583–591.

[6] M. Karpiriski, A. Senart, and V. Cahill, "Sensor networks for smart roads," in *Proc. 4th Annu. IEEE Int. Conf. Pervasive Comput. Commun. Workshops*, Mar. 2006, pp. 306–310.

[7] T. He, S. Krishnamurthy, J. A. Stankovic, T. Abdelzaher, L. Luo, R. Stoleru, T. Yan, L. Gu, J. Hui, and B. Krogh, "Energy-efficient surveillance system using wireless sensor networks," in *Proc. 2nd Int. Conf. Mobile Syst., Appl., Serv., MobiSys*, 2004, pp. 270–283.

[8] G. Simon, M. Maróti, Á. Lédeczi, G. Balogh, B. Kusy, A. Nádas, G. Pap, J. Sallai, and K. Frampton, "Sensor network-based countersniper system," in *Proc. 2nd Int. Conf. Embedded Netw. Sens. Syst., SenSys*, 2004, pp. 1–12.

[9] J. Stankovic, T. Abdelzaher, C. Lu, L. Sha, and J. Hou, "Real-time communication and coordination in embedded sensor networks," *Proc. IEEE*, vol. 91, no. 7, pp. 1002–1022, Jul. 2003.

[10] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "An application-specific protocol architecture for wireless microsensor networks," *IEEE Trans. Wireless Commun.*, vol. 1, no. 4, pp. 660–670, Oct. 2002.

[11] J. Polastre, J. Hill, and D. Culler, "Versatile low power media access for wireless sensor networks," in *Proc. 2nd Int. Conf. Embedded Netw. Sens. Syst., SenSys*, 2004, pp. 95–107.

[12] V. P. Mhatre, C. Rosenberg, D. Kofman, R. Mazumdar, and N. Shroff, "A minimum cost heterogeneous sensor network with a lifetime constraint," *IEEE Trans. Mobile Comput.*, vol. 4, no. 1, pp. 4–15, Jan./Feb. 2005.

[13] O. Younis and S. Fahmy, "HEED: A hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks," *IEEE Trans. Mobile Comput.*, vol. 3, no. 4, pp. 366–379, Oct.–Dec. 2004.

[14] F. Kuhn, T. Moscibroda, and R. Wattenhofer, "Initializing newly deployed ad hoc and sensor networks," in *Proc. 10th Annu. Int. Conf. Mobile Comput. Netw., MobiCom*, 2004, pp. 260–274.

[15] W. Yu, J. Cao, X. Zhou, X. Wang, K. C. C. Chan, A. T. S. Chan, and H. V. Leong, "A high-throughput mac protocol for wireless ad hoc networks," *IEEE Trans. Wireless Commun.*, vol. 7, no. 1, pp. 135–145, Jan. 2008.

[16] Q. Han, S. Mehrotra, and N. Venkatasubramanian, "Application-aware integration of data collection and power management in wireless sensor networks," *J. Parallel Distrib. Comput.*, vol. 67, no. 9, pp. 992–1006, Sep. 2007.

[17] L. Paradis and Q. Han, "TIGRA: Timely sensor data collection using distributed graph coloring," in *Proc. 6th Annu. IEEE Int. Conf. PERCOM*, 2008, pp. 264–268.

[18] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 2nd ed. Cambridge, MA: MIT Press, 2006.

[19] K. Akkaya and M. Younis, "C2ap: Coverage-aware and connectivity-constrained actor positioning in wireless sensor and actor networks," in *Proc. IPCCC*, Apr. 2007, pp. 281–288.

[20] B. McLaughlan and K. Akkaya, "Coverage-based clustering of wireless sensor and actor networks," in *Proc. IEEE ICPS*, Jul. 2007, pp. 45–54.

[21] T. He, J. A. Stankovic, C. Lu, and T. Abdelzaher, "SPEED: A stateless protocol for real-time communication in sensor networks," in *Proc. 23rd ICDCS*, 2003, pp. 46–55.

[22] S. Doshi, S. Bhandare, and T. X. Brown, "An on-demand minimum energy routing protocol for a wireless ad hoc network," *SIGMOBILE Mobile Comput. Commun. Rev.*, vol. 6, no. 3, pp. 50–66, Jul. 2002.

[23] C. Intanagonwiwat, R. Govindan, D. Estrin, J. Heidemann, and F. Silva, "Directed diffusion for wireless sensor networking," *IEEE/ACM Trans. Netw.*, vol. 11, no. 1, pp. 2–16, Feb. 2003.

[24] M. Ègalj, J.-P. Hubaux, and C. Enz, "Minimum-energy broadcast in all-wireless networks: Np-completeness and distribution issues," in *Proc. 8th Annu. Int. Conf. Mobile Comput. Netw., MobiCom*, 2002, pp. 172–182.

[25] A. Manjeshwar and D. Agrawal, "APTEEN: A hybrid protocol for efficient routing and comprehensive information retrieval in wireless sensor networks," in *Proc. IPDPS*, 2002, pp. 195–202.

[26] J. E. Wieselthier, G. D. Nguyen, and A. Ephremides, "Energy-efficient broadcast and multicast trees in wireless networks," *Mobile Netw. Appl.*, vol. 7, no. 6, pp. 481–492, Dec. 2002.

[27] W. Ye, J. Heidemann, and D. Estrin, "An energy-efficient MAC protocol for wireless sensor networks," in *Proc. IEEE INFOCOM*, Jun. 2002, pp. 1567–1576.

[28] T. van Dam and K. Langendoen, "An adaptive energy-efficient MAC protocol for wireless sensor networks," in *Proc. 1st Int. Conf. Embedded Netw. Sens. Syst., SenSys*, 2003, pp. 171–180.

[29] G. Lu, B. Krishnamachari, and C. S. Raghavendra, "An adaptive energy-efficient and low-latency MAC for tree-based data gathering in sensor networks: Research articles," *Wireless Commun. Mobile Comput.*, vol. 7, no. 7, pp. 863–875, Sep. 2007.

[30] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris, "Span: An energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks," *Wireless Netw.*, vol. 8, no. 5, pp. 481–494, Sep. 2002.

[31] G. Xing, C. Lu, Y. Zhang, Q. Huang, and R. Pless, "Minimum power configuration in wireless sensor networks," in *Proc. 6th ACM Int. Symp. Mobile Ad Hoc Netw. Comput., MobiHoc*, 2005, pp. 390–401.

Wenzhong Li (S'05–M'08) received the B.S. and Ph.D. degrees in computer science from Nanjing University, Nanjing, China, in 2002 and 2007, respectively.

He is currently an Assistant Professor with the Department of Computer Science and Technology, Nanjing University. He has published more than 30 research papers at international conferences and journals, which include the International Conference on Distributed Computing Systems, the International Workshop on Quality of Service, the International Conference on Parallel Processing, the International Conference on Communications, the IEEE Wireless Communications and Networking Conference, *Computer Communications*, and the *Journal of Supercomputing*. His research interests include wireless networks, sensor networks, pervasive computing, and peer-to-peer computing.

Dr. Li is a member of the Association for Computing Machinery. He has published papers in the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY. He was the recipient of the Best Paper Award at the 2009 ICC.

Edward Chan (M'88) received the B.Sc. and M.Sc. degrees in electrical engineering from Stanford University, Stanford, CA, and the Ph.D. degree in computer science from Sunderland University, Sunderland, U.K.

He worked in industry for a number of years in the design and implementation of computer networks and real-time control systems before joining City University of Hong Kong, Kowloon, Hong Kong, where he is currently an Associate Professor with the Department of Computer Science. His current research interests include performance evaluation of high-speed networks, mobile data management, power-aware computing, and network management.

Mounir Hamdi (S'90–M'91–SM'06) received the B.S. degree in electrical engineering (computer engineering option with distinction) from the University of Louisiana, Lafayette, in 1985 and the M.S. and Ph.D. degrees in electrical engineering from the University of Pittsburgh, Pittsburgh, PA, in 1987 and 1991, respectively.

From 1985 to 1991, he was a Teaching/Research Fellow with the Department of Electrical Engineering, University of Pittsburgh, where he was involved in major research projects, as well as teaching undergraduate courses. Since 1991, he has been a faculty member with the Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Kowloon, Hong Kong, where he is currently Head and Professor of Computer Science and Engineering. He has been the Director of the Computer Engineering Program that has about 350 undergraduate students for nine years and the Director of the Master of Science in Information Technology that has about 150 postgraduate students for three years. He is also the Director of the Computer Engineering Laboratory. From 1999 to 2000, he held Visiting Professor positions at Stanford University, Stanford, CA, and the Swiss Federal Institute of Technology, Lausanne, Switzerland. His general areas of research are high-speed packet switches/routers, optical networks, and wireless local area networks, in which he has published more than 260 research publications, received numerous research grants, and supervised some 30 postgraduate students. His current research interests are high-speed networks, including the design, analysis, scheduling, and management of high-speed switches/routers, wavelength division multiplexing networks/switches, and wireless networks.

Dr. Hamdi is a Senior Member of the Association for Computing Machinery.

Sanglu Lu (M'04) received the B.S., M.S., and Ph.D. degrees in computer science from Nanjing University, Nanjing, China, in 1992, 1995, and 1997, respectively.

She is currently a Professor with the Department of Computer Science and Technology and the State Key Laboratory for Novel Software Technology, Nanjing University. Her research interests include distributed computing, pervasive computing, and wireless networks.

Daoxu Chen (M'04) received the B.S. degree in computer science from Nanjing University, Nanjing, China, in 1982.

He has been a Visiting Researcher with Purdue University, West Lafayette, IN, and City University of Hong Kong, Kowloon, Hong Kong. He is currently a Professor with the Department of Computer Science and Technology, Nanjing University, Nanjing, China. His research interests include distributed computing, parallel processing, and computer networks.

Prof. Chen is a member of the Association for Computing Machinery and a Senior Member of the China Computer Federation.