An Adaptive Delay-Minimized Route Design for Wireless Sensor–Actuator Networks

Edith C.-H. Ngai, Jiangchuan Liu, *Senior Member*, and Michael R. Lyu

***Abstract*—Wireless sensor–actuator networks (WSANs) have recently been suggested as an extension to conventional sensor networks. The powerful and mobile actuators can patrol along different routes and communicate with the static sensor nodes. Obviously, it is crucial to optimize the routes for the actuators to collect the sensor data in a timely fashion. Given the nonuniform and time-varying distribution of sensors and events in large networks, the route design has to be dynamic and scalable as well as balance the loads of the actuators. In this paper, we propose probabilistic route design (PROUD), which is an effective and adaptive algorithm for weight-differentiated route calculation. PROUD constructs an *a priori* route that covers the sensor locations, following which, the actuators probabilistically and cyclically visit the sensor locations according to their weights. We show that this probabilistic approach adapts well to network dynamics without frequent recalculation of the whole route. It works for both small-scale sensor–actuator networks and large-scale sensor–actuator networks with partitioning. We further develop a distributed implementation of PROUD and extend it to accommodate actuators with variable speeds. Finally, we devise a multiroute improvement and a task-exchange algorithm that enable load balancing. Our performance evaluation shows that PROUD effectively reduces the overall data-collection time and evenly distributes the energy consumption across the actuators, as compared with other state-of-the-art solutions.**

***Index Terms*—Actuators, route design, wireless sensor networks (WSNs).**

I. Introduction

WIRELESS sensor networks (WSNs) have been applied in a broad spectrum of applications, ranging from environment monitoring and target tracking to battlefield surveillance and chemical attack detection [1]–[4]. The asymmetric communication patterns from the sensors to the sink, however, often overload the sensors close to the sinks and consequently reduce the network lifetime. Moreover, network partitions may occur in sensor networks, which make multihop communications impossible. To alleviate these problems, mobile elements such as mobile sinks [5] or mobile relays [6] have been suggested for collecting data in WSNs. Actuators, which have stronger computation and communication power than unipurpose microsensors, have also been introduced [7], [8]. In a wireless sensor–actuator network (WSAN), a mobile actuator can move around to cover the sensing field and interact with static sensors. Each static sensor maintains a size-limited buffer that temporarily stores the sensed data until some actuator approaches; it then uploads the data to the actuator with short-range communications and frees the buffer [9], [10].

The amount and frequency of data generation across a sensing field are, in general, nonuniform [11]. The sensors with higher data generation rate or the locations with higher event-occurring probability naturally expect more frequent visits. More formally, there is a route design problem (RDP) for the actuators to minimize their average interarrival time to the static sensors [12]. Given that the weight of sensors and event frequency are time varying, an adaptive solution is expected. For a large-scale sensor network with multiple actuators, a distributed and load-balanced implementation is also necessary.

In this paper, we propose probabilistic route design (PROUD), which is an effective and adaptive algorithm for weight-differentiated route calculation. PROUD constructs an *a priori* route that covers the sensor locations, following which, the actuators probabilistically and cyclically visit the sensor locations according to their weights. This probabilistic approach with prior route adapts well to network dynamics without frequent recalculation of the whole route. It works for both small-scale sensor–actuator networks and large-scale networks with partitioning. We further develop a distributed implementation of PROUD and extend it to accommodate actuators with variable speeds, targeting applications with bounded interarrival time demand. Finally, we devise a multiroute improvement and a task-exchange algorithm to provide load balancing to the actuators. Our simulation results show that PROUD can effectively reduce the overall data collection time and evenly distribute the energy consumption across the actuators.

The remainder of this paper is organized as follows. The related work is presented in Section II, followed by an overview of the RDP in Section III. The PROUD algorithm is described in Section IV, and a distributed implementation is shown in Section V. In Section VI, we discuss possible enhancements for integrating actuators with variable speeds and balancing their workloads. Simulation results are presented in Section VII. Finally, Section VIII concludes the paper.

Manuscript received October 15, 2007; revised December 6, 2008. First published June 2, 2009; current version published November 11, 2009. This work was supported in part by the Research Grants Council of the Hong Kong Special Administrative Region, China, under Project CUHK4158/08E. The work of E. C.-H. Ngai was supported by the Uppsala VINN Excellence Center for Wireless Sensor Networks, which is supported by VINNOVA, Sweden. The work of J. Liu was supported by the Natural Sciences and Engineering Research Council of Canada under a Discovery Grant and a Strategic Project Grant. The review of this paper was coordinated by C. Lin.

E. C.-H. Ngai is with the Department of Information Technology, Uppsala University, 751 05 Uppsala, Sweden (e-mail: edith.ngai@it.uu.se).

J. Liu is with the School of Computing Science, Simon Fraser University, Vancouver, BC V5A 1S6, Canada (e-mail: jcliu@cs.sfu.ca).

M. R. Lyu is with the Department of Computer Science and Engineering, Chinese University of Hong Kong, Shatin, Hong Kong (e-mail: lyu@cse.cuhk.edu.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.2009.2024155

0018-9545/$26.00 © 2009 IEEE

II. Related Work

Recently, mobile elements have been suggested for assisting data delivery in diverse wireless networks. Pathirana *et al.* [13] examined the location estimation and trajectory predication for mobile base stations with a focus on cellular networks. Zhao *et al.* [14] proposed a *message ferrying* approach to address the network partition problem in sparse ad hoc networks. Somasundara *et al.* [11] further formulated the problem of scheduling the mobile element with deadlines and presented earliest deadline- and minimum-weight-based heuristics.

For sensor networks, Shah *et al.* [15] presented an architecture using moving entities (*data mules*) to collect sensing data. There have also been studies on mobile sinks with predictable and controllable movement patterns [16], [17] and the optimal time schedule for locating sojourn points [5]. Luo and Hubaux [6] further investigated a joint mobility and routing algorithm with mobile relays to prolong the lifetime of WSNs. Gu *et al.* [9] proposed a partitioning-based algorithm to schedule the movement of mobile elements, which minimizes the required moving speed and eliminates buffer overflow. Their solution was customized for an "Eye" topology, where the events are concentrated at certain locations. Solutions for sensor networks with general distributions remain to be explored. Recently, Bisnik *et al.* [18] studied the problem of providing quality coverage using mobile sensors and analyzed the effect of controlled mobility on the fraction of events captured. Their focus, however, is not on the route design.

Our work is motivated by the above studies. The key difference is that we focus on adaptive and distributed route design for multiple mobile elements in WSANs, specifically, actuators moving along independent routes. We also address the nonuniform weights of the static sensors with a novel probabilistic solution.

Route design for mobile elements has also been studied in delay-tolerant networks [19], but the target is on point-to-point data transfer. The vehicle routing problem (VRP) is another related problem, which considers scheduling vehicles stationed at a central facility to support customers with known demands [20]. There are a number of variations to VRP, e.g., capacitated VRP [21] and VRP with time windows [22], yet the unique characteristics of actuators and the heterogeneity of sensor weights make the route design in WSANs different.

III. Overview of the RDP

We consider a WSAN consisting of M mobile actuators and N static sensors. Each of the sensors and actuators is equipped with a wireless transceiver. The actuators move in the sensing field along independent routes, at constant or variable speeds. Each static sensor maintains a size-limited buffer to temporarily store the sensed data. When an actuator approaches, the sensor uploads the data to the actuator and frees the buffer. The sensors may have different weights related to their data generation rates or event-occurring frequencies, which may also change over time.

The routes of the actuators should be designed to minimize the expected delay for data uploading, and intuitively, the sensors with higher weights expect shorter average actuator interarrival times. Formally speaking, the RDP strikes to minimize the weighted average actuator interarrival time to sensors, that is

$$Minimize \sum_{\forall i} A_i w_i N_i \tag{1}$$

where A_i and N_i are the actuator interarrival time and the total number of sensors with weight w_i, respectively. We focus on cyclical routes that starts from and ends at the same location, and hence, only the optimal route of each cycle needs to be calculated.

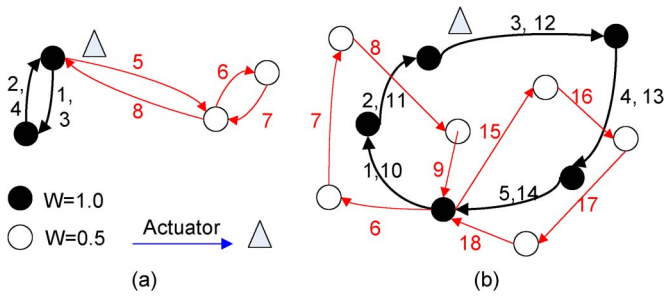

Fig. 1. Two examples of route design with a single actuator, where the visiting sequence along the route is marked next to the edges.

Fig. 1 illustrates two examples of route design in a single-actuator case. The set of black nodes S_b and the set of white nodes S_w have weights of $W_b = 1.0$ and $W_w = 0.5$, respectively. Let A_b and A_w be the respective actuator interarrival times of all black and white nodes. Assuming that actuators move at a constant speed, obviously, we expect that the interarrival time of S_b will be half of S_w, such that $A_w = 2A_b$. As illustrated in Fig. 1(a), the actuator will visit the black nodes twice and the white nodes once every cycle. The average interarrival time of white nodes is thus

$$A_w = T(S_b) + T(S_w) + 2t(S_b, S_w) \tag{2}$$

where $T(S_b)$ is shortest possible travel time taken to visit the set of nodes S_b in one cycle, and $t(S_b, S_w)$ is the shortest travel time for the actuator to walk between the sets S_b and S_w. The travel time between two nodes depends on the landscapes, obstacles, and moving speed of the actuator. Our problem formulation generally applies to networks deployed over diverse terrains and with actuators of various types of engines.

The shortest possible travel times $T(S_b)$ and $T(S_w)$ can be modeled as the lowest possible cost in the traveling-salesman problem (TSP). Note that the TSP itself is an NP-complete problem but with fast and bounded approximation algorithms [23]–[26]. Fig. 1(b) shows a more complicated example. Again, it is easy to see that the RDP is NP-hard, even in this single-actuator case.

In practice, we may also model the cost between two nodes as any nondecreasing function of their distance, instead of the travel time taken by individual actuators.

IV. PROUD Algorithm

We now describe our PROUD algorithm. In PROUD, an *a priori* route is calculated during network initialization; the

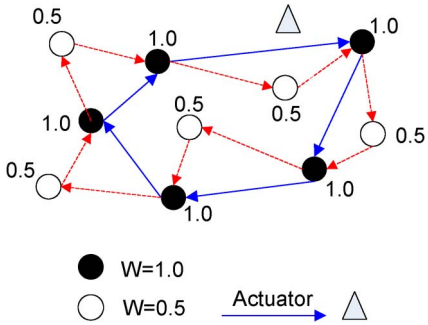

Fig. 2. Probabilistically visiting nodes according to their weights.

sensors are then probabilistically visited along the route according to their weights. For instance, sensors with a visiting probability of 1.0 are visited in every cycle, while sensors with a visiting probability of 0.5 only have half a chance to be visited in each cycle. By resetting the visiting probabilities, an actuator can easily update the interarrival times based on data generation rates or event-occurring frequencies. As such, the network dynamics can be accommodated without frequently recalculating the whole route.

In the following, we first give a centralized design that is executed by one particular actuator or the base station. We will extend it to a distributed implementation in the next section.

A. Small-Scale Networks With No Partitioning

1) Forming an A Priori Route: An *a priori* route is formed by constructing a TSP path that contains all locations to be visited. We adopt the well-known Approx-TSP-Tour algorithm [26] here for its low cost and bounded performance. This algorithm first creates a minimum spanning tree (MST) [26], whose weight is a lower bound on the length of an optimal traveling-salesman tour. It then creates a tour based on the MST, with the cost being no more than twice that of the optimal. Both calculations are done in polynomial time.

2) Probabilistically Visiting Sensors: We then apply a probabilistic visiting model, in which an actuator sequentially but selectively visits related sensors along the *a priori* route. Let $s_1, s_2, \ldots, s_i, s_{i+1}, \ldots, s_n$ be the sequence of sensor locations along the *a priori* route. After visiting location s_i, the actuator determines whether to visit s_{i+1} by generating a random number between 0 and 1. If the random number is smaller than the visiting probability of s_{i+1}, i.e., p_{i+1}, then it visits s_{i+1} in the next step. If not, the actuator skips s_{i+1} and determines whether to visit the next location s_{i+2}, and so forth.

Intuitively, the sensors with higher weights should be assigned with a higher probability such that they are visited more frequently. Hence, we set the visiting probability p_i of a location i to be w_i, where w_i is the (normalized) weight of the sensors. Fig. 2 shows an example of PROUD with two types of sensor nodes. The black nodes have a visiting probability of 1.0, which indicates that they will be visited in every cycle. On the other hand, each white node is visited only with a probability of 0.5 in every cycle.

3) Allocating the Actuators: For a small network with no partitioning, the actuators can evenly be placed along the *a priori* route during initialization. The expected route length with probabilistic visiting can be calculated by

$$E[R] = \sum_{r=0}^{n-2} \sum_{i=1}^{n} C(i, i+r) p_i p_{i+r+1} \Pi_{k=1}^{r} (1 - p_{i+k}) \quad (3)$$

where $1, \ldots, n$ is a sequence of nodes on the route R, $C(i, j)$ is the travel cost between i and j, and p_i is the visiting probability of i. Note that $E[R]$ depends not only on the visiting probability but also on the network topology. For instance, an actuator may have to visit several sensor nodes before visiting a particular one if these nodes are all located along the same segment.

For a sensor i with a visiting probability p_i, its average actuator interarrival time A_i is thus

$$A_i = \frac{E[R]}{p_i v M} \quad (4)$$

where v is the average moving speed of the actuators.

In a dynamic environment, the visiting probability of the sensors can be updated according to their data generation rate or event frequency, but the route does not have to be recalculated for each individual change.

Time Complexity Analysis: Recall that N and M are the number of sensors and the number of actuators, respectively. The running times of the steps in the PROUD algorithm are then given as follows.

- Step 1: The running time of the Approx-TSP-Tour algorithm is $O(E) = O(N^2)$, since the input is a complete graph.
- Step 2: The time is $O(N)$ for an actuator to select the next locations according to the visiting probability in every cycle.
- Step 3: The time of actuator allocation is $O(M)$.

In summary, the PROUD algorithm has an overall time complexity of $O(N^2 + M)$.

Bound Analysis: Since the interarrival time A_i is proportional to the weights of sensors, we can focus on analyzing the A_i of the locations in the lowest weight range. Let A_i and A_i^* be the average actuator interarrival time for sensors S_i in the lowest weight range w_i in PROUD and the optimal algorithm, respectively. The optimal algorithm would visit all locations in the lowest weight range at least once in a cycle. Thus, the actuator will walk along a route with a length of at least $|\text{TSP}(S_i)|$. Since there are M actuators in the network, we have

$$A_i^* \geq \frac{|\text{TSP}(S_i)|}{vM} \quad (5)$$

which gives the lower bound of the optimal solution.

The ratio of A_i/A_i^* is equal to

$$\frac{A_i}{A_i^*} \leq \frac{E[R]}{p_i \, |\text{TSP}(S_i)|}. \quad (6)$$

From (4), the interarrival time A_i for the sensors in the weight range i in PROUD depends on the expected route length

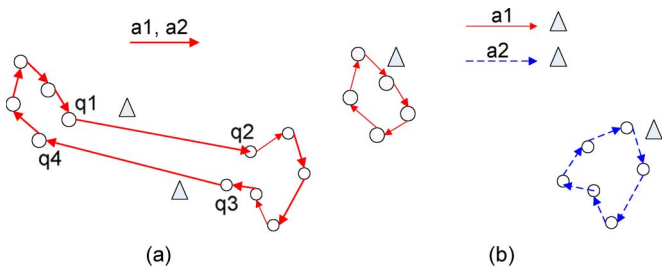

Fig. 3. Two actuators walking along (a) the same route and (b) distinct routes.

$E[R]$. Since an actuator visits only a subset of the sensors along the prior route in each cycle, the expected route length $E[R]$ cannot be longer than the prior route constructed by the TSP path that contains all the sensors S in the network, i.e., $E[R] \leq |\text{TSP}(S)|$. The interarrival time A_i of the sensors in the weight range i is thus bounded by

$$A_i \leq \frac{|\text{TSP}(S)|}{p_i v M}. \tag{7}$$

B. Large-Scale Network With Partitioning

In large-scale sensor networks, network partitions may happen, dividing the sensors into different clusters. In this case, letting actuators share the same route may not be as efficient as walking along distinct routes. Consider the network in Fig. 3, where the route designs with two actuators walking on the same route and distinct routes are depicted in Fig. 3(a) and (b), respectively. Clearly, the routes in Fig. 3(b) can achieve shorter interarrival times than those in Fig. 3(b) if

$$C(q1, q4) + C(q2, q3) \leq C(q1, q2) + C(q3, q4). \tag{8}$$

This suggests that the sensor distribution should be an important consideration on route design. In particular, the sensors in different clusters should be visited by actuators along independent routes to minimize the interarrival time.

1) Forming Clusters: We use a recursive algorithm for clustering the sensors, as shown in Algorithm 1. In each recursion, it divides the MST into two subtrees by removing its longest edge e, provided that $w(e)/w(m) \geq \delta$, where $w(e)$ is the cost of edge e. By doing this, the sensors that are geographically far away will be involved in different subtrees and, later, distinct routes. Note that δ is set to ensure that the number of clusters is smaller than the number of actuators.

Algorithm 1 Clustering the sensors
$Function$ Cluster(MST(S))
Find the edge m with the median length;
Find the longest edge e;
if $w(e)/w(m) \geq \delta$ **then**
delete edge e;
Cluster(MST(S_1));
Cluster(MST(S_2));
end if

2) Forming A Priori Routes and Probabilistically Visiting Sensors: After clustering the sensors, the PROUD algorithm can be applied in each cluster following the simple case of small-scale networks.

3) Allocating the Actuators: Multiple routes are formed from the above. They may have different expected route lengths due to the heterogeneous sensor locations and visiting probabilities in the clusters. The uneven expected route lengths may cause unequal interarrival times for the sensors with the same weight. To address this problem, we allocate different numbers of actuators to the routes. Intuitively, routes with longer expected lengths should be allocated with more actuators. This is illustrated in Algorithm 2, where N_R is the total number of routes, $remain_a$ is the number of remaining unassigned actuators, and n_j is the number of actuators assigned to route R_j.

Algorithm 2 Actuator allocation for distinct routes
for $j = 1$ to N_R **do**
$n_j = 1$;
end for
$remain_a = M - N_R$;
while $remain_a > 0$ **do**
Find the maximum $E[R_j^*]$;
$E[R_j^*] = E[R_j^*]n_j^*/(n_j^* + 1)$;
$n_j^* + +$;
$remain_a - -$;
end while

V. Distributed Implementation

For large-scale networks, it can be difficult for a single node to collect the information and execute the route design algorithm in a centralized manner. To this end, we next present a practical distributed implementation for PROUD, in which sensors and actuators form clusters by cooperatively constructing MSTs.

A. Forming R-Clusters

First, the sensors locally construct MSTs by communicating with their neighbors. Given the communication range of sensors, i.e., R_s, the weight of each edge e in the MST must be smaller than or equal to R_s, that is, $w(e) \leq R_s$. We refer to such an MST as an *R-Cluster*, i.e., $RC(V, E)$. The cost of the R-cluster is denoted by $Cost(RC)$, which is the sum of $w(e)$, $\forall e \in E$. It will be stored by the sensors in $RC(V, E)$. There are many existing distributed algorithms for forming an MST [27], [28], and we apply a fast algorithm from [29] for this purpose.

B. Connecting R-Clusters

An R-cluster forest is formed by the sensors as above. These R-clusters can be connected together to form MSTs that contain more sensor locations. We divide the network into M subareas, each of which is explored by one actuator. Each actuator looks for the R-clusters in its area and connects them if they are within a certain distance, e.g., $C(RC_1, RC_2) \leq \delta$. Then, a new cluster is formed with cost $Cost(RC_1) + Cost(RC_2) + C(RC_1, RC_2)$.

Similarly, the actuators also connect their R-clusters/clusters with those in their neighboring areas. Algorithm 3 shows how two actuators $A1$ and $A2$ connect their R-clusters RC_1 and RC_2, where BD is the boundary of the two corresponding areas.

Algorithm 3 Connecting the R-Clusters
$Function$ Connect-Cluster$(RC_1(V_1, E_1), RC_2(V_2, E_2))$
if $(C(RC_1, BD) \leq \delta)$ and $(C(RC_2, BD) \leq \delta)$ **then**
Actuators A_1 and A_2 exchange locations close to BD;
Find the shortest edge e that connects RC_1 and RC_2;
if $w(e) \leq \delta$ **then**
Form new cluster $\mathcal{C}_{\text{new}}(V, E)$;
$V = V_1 \bigcup V_2$;
$E = E_1 \bigcup E_2 \bigcup \{e\}$;
$Cost(\mathcal{C}_{\text{new}}) = Cost(RC_1) + Cost(RC_2) + w(e)$;
end if
end if

C. Allocating Actuators

Then, the actuators are to be allocated to the clusters, such that each cluster is served by at least one actuator. Each actuator associates itself to any unassigned clusters in its area. If the associated cluster is crossing two or more areas, the actuator has to inform the actuators in those areas. It is possible that the number of clusters is greater than the number of actuators. The unassigned clusters can be connected with some assigned clusters to ensure that they are served by at least one actuator. On the contrary, a remaining actuator can associate itself with a nearby cluster with the highest cost. If multiple actuators are serving one cluster, they can equally divide it and independently serve the sensors involved. Finally, an *a priori* route is computed by the actuator in each cluster using the Approx-TSP-Tour algorithm [26].

VI. Enhancements to PROUD

So far, we have considered actuators with constant speeds only. We next explore actuators with variable speeds to further reduce the interarrival time for heterogeneous networks. We also present two enhancements for load balancing among actuators.

A. Actuators With Variable Speeds

Let o_i be the expected average actuator interarrival time for the sensors with weight w_i. The highly weighted sensors intuitively have shorter expected average actuator interarrival times than the others, i.e., $o_1 < o_2 \ldots < o_i < \ldots o_m$, where o_m is the expected average actuator interarrival time of the least weighted sensors. To achieve this, the highly weighted sensors will be assigned with higher visiting probability. For simplicity, we normalize the visiting probability p_i for the sensors according to their expected average actuator interarrival time o_i. We set the visiting probability $p_1 = 1$ for the sensors with the shortest expected average actuator interarrival time o_1. The visiting probability p_i of the remaining sensors with the expected average actuator interarrival time, e.g., o_i, are calculated by $p_i = o_1/o_i$. The visiting probability to sensors can adaptively be updated by the actuators according to the dynamic change of the expected average actuator interarrival time. By adjusting the speeds of the actuators, we can ensure that sensors with the same visiting probability achieve similar interarrival times, even if they are visited by different actuators along distinct routes.

Assume that node i on R_j has a probability p_i to be visited by actuator j every cycle. Its average actuator interarrival time A_i can be calculated as

$$A_i = E[R_j]/p_i v_j \tag{9}$$

where v_j is the moving speed of actuator j.

Given the expected average actuator interarrival time o_i, we can calculate the minimum moving speed of the actuator to satisfy this requirement, i.e., $A_i \leq o_i$.

From (9), we obtain

$$E[R_j]/p_i v_j \leq o_i. \tag{10}$$

Note that $E[R_j]$, p_i, v_j, and o_i are all greater than zero. From (10), we will get the minimum moving speed as

$$v_j \geq E[R_j]/p_i o_i. \tag{11}$$

Without loss of generality, v_j can easily be determined by assuming $p_i = 1$, that is, $v_j \geq E[R_j]/o_1$.

B. Load Balancing in Route Design

Since the energy consumption of mobile actuators increases with their speeds [30], the unequal moving speeds might cause imbalanced energy consumption. To tackle this problem, we propose two algorithms for balancing load across the actuators, which still retain the energy efficiency of the route design.

1) Multiroute Improvement Algorithm: Since the actuator having a longer route consumes more energy, the loads of actuators can be balanced by forming routes with identical expected lengths. To this end, a loaded actuator may assign some of its sensor locations to its neighboring actuator with the minimum expected route length.

Consider two routes R_1 and R_2 involved in the multiroute improvement. Their new expected route lengths become ideal if $E[R_1'] = E[R_2'] = (E[R_1] + E[R_2])/2$. In other words, R_1 should transfer a length of $(E[R_1] - E[R_2])/2$ to R_2. Although the sensor locations can be transferred one by one from $R1$ to $R2$, until the expected lengths of the two routes become equal, the serial operation can be quite inefficient. Hence, we provide a fast approximation to find the proportion of sensor locations ξ to be transferred from MST_1 to MST_2

$$\frac{cost(\xi)}{cost(\text{MST}_1)} = \frac{(E[R_1] - E[R_2])/2}{E[R_1]} \tag{12}$$

where $cost(\xi)$ and $cost(\text{MST}_1)$ represent the costs of the MSTs that contain the sensor locations in ξ and R_1, respectively.

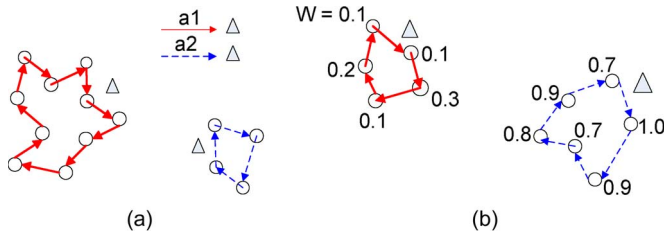

Fig. 4. Routes involving (a) different amount of sensors and (b) sensors with different weights.

TABLE I
SIMULATION PARAMETERS

Network size	200m x 200m
Sensor distribution	Uniform random or Cluster-based uniform or Cluster-based non-uniform
No. of sensors (N)	100
Weight of sensors (W_j)	0.0-1.0
No. of actuators	M
Radio range	40m
Packet size	32bytes
Transmit data rate	76.8kbps
Radio receiver current	9.6mA
Radio transmitter current	16.5mA
Supply voltage	3V
Actuator moving speed	v
Motion power	$p(v)$

2) Task-Exchange Algorithm: In a network involving clusters with different sizes or weights, it can be more efficient for one actuator to take up more load than another. For example, the two actuators walking along two distinct routes with unequal lengths (see Fig. 4) achieve better performance than those along routes with identical lengths. In this case, enforcing load balancing by equalizing the lengths of the two routes may not be the best choice. Instead, load balancing among the actuators can be achieved by exchanging their routes.

Intuitively, an overloaded actuator may exchange its route with another actuator traveling at a lower speed. More formally, we define $Energy_{A1}$ and $Energy_{A2}$ to be the remaining energy of actuators $A1$ and $A2$ and v_1 and v_2 to be the minimum actuator speeds on routes $R1$ and $R2$. A task-exchange algorithm is executed when actuator $A1$ has less remaining energy than $A2$, but it requires a higher moving speed. The tasks of the two actuators are exchanged by swapping their routes. By doing this, $A1$ can walk on a shorter route at a lower speed and reduce its energy consumption. On the contrary, $A2$ consumes more energy with a higher moving speed, but load balancing is achieved as it has more energy than $A1$.

VII. PERFORMANCE EVALUATION

We have conducted extensive simulations to evaluate our proposed PROUD algorithm and to compare it with state-of-the-art solutions. Unless otherwise specified, the network configurations summarized in Table I are used in our simulations. The configurations are mainly drawn from existing works [8], [31], [32]. The energy consumption for communications is based on the CC1000 RF transceiver [33] in the widely used MICA2 Motes [34]–[36]. We also used four typical network topologies in our simulations to comprehensively examine the algorithms, including the uniform sensor distribution, the "Eye" topology, the cluster-based uniform sensor distribution, and the cluster-based nonuniform sensor distribution.

A. Average Actuator Interarrival Time

In the first set of experiments, we evaluate the average actuator interarrival time A_{avg} under the series of typical sensor distributions with the average moving speed of the actuator at 1 m/s.

We also compare PROUD with two state-of-the-art algorithms: the *partitioning-based scheduling* (PBS) algorithm [9] and the *bounded event loss probability* in the 2-D space (BELP-2D) algorithm [18]. The PBS algorithm partitions all nodes into several groups (called *bins*) and forms a schedule that concatenates them such that buffer overflow can be avoided in sensors with different data-generation rates. The BELP-2D algorithm deals with the bounded event loss problem in a 2-D space, which ensures that the time that elapsed between two consecutive visits is less than a critical time. It uses the solutions of the TSP with neighborhoods to find routes. To achieve a fair comparison, we adopt the Approx-TSP-Tour algorithm [26] to approximate the TSP paths in all the three algorithms.

1) Uniform Random Sensor Distribution: Fig. 5(a) shows the average interarrival time A_{avg} for an actuator to periodically visit the sensors under uniform random sensor distribution with $N = 100$ and $M = 5$. It evaluates the interarrival times A_{avg} to the sensors with weights in the ranges 0.0–0.2, 0.2–0.4, 0.4–0.6, 0.6–0.8, and 0.8–1.0, respectively.

The results demonstrate that PROUD, PBS, and BELP-2D have comparable interarrival times A_{avg} for sensors with $w=1$. Both PROUD and PBS differentiate the actuator interarrival times according to the weights of sensors. The sensors with higher weights achieve shorter interarrival times A_{avg}. However, the A_{avg} of PBS is impractically long for most sensors with lower weights. This is simply because the locations of bins are widely spread under uniform random sensor distribution.

On the other hand, BELP-2D constantly achieves a low A_{avg} for all sensors, although it does not differentiate the interarrival times at all. This is because the route in BELP-2D is the shortest TSP path that contains all the sensor locations. Nevertheless, the A_{avg} of sensors with $w = 1$ in PROUD is slightly lower than that in BELP-2D. PROUD is still more suitable for sensor networks with different weights as it can satisfy a shorter A_{avg} requirement for sensors with $w = 1$.

2) "Eye" Topology: Next, we evaluate our algorithm under the "Eye" topology [9]. In this topology, events are concentrated at the center of the network. A sequence of concentric circles divides the network area into several ring-shaped regions. The sensors in the innermost region are assigned with the highest weight. The weights decrease for sensors in the radially outward regions.

Fig. 5(b) shows that PBS performs pretty well under this particular "Eye" topology. It achieves a shorter A_{avg} than both PROUD and BELP-2D for highly weighted sensors. Its A_{avg}

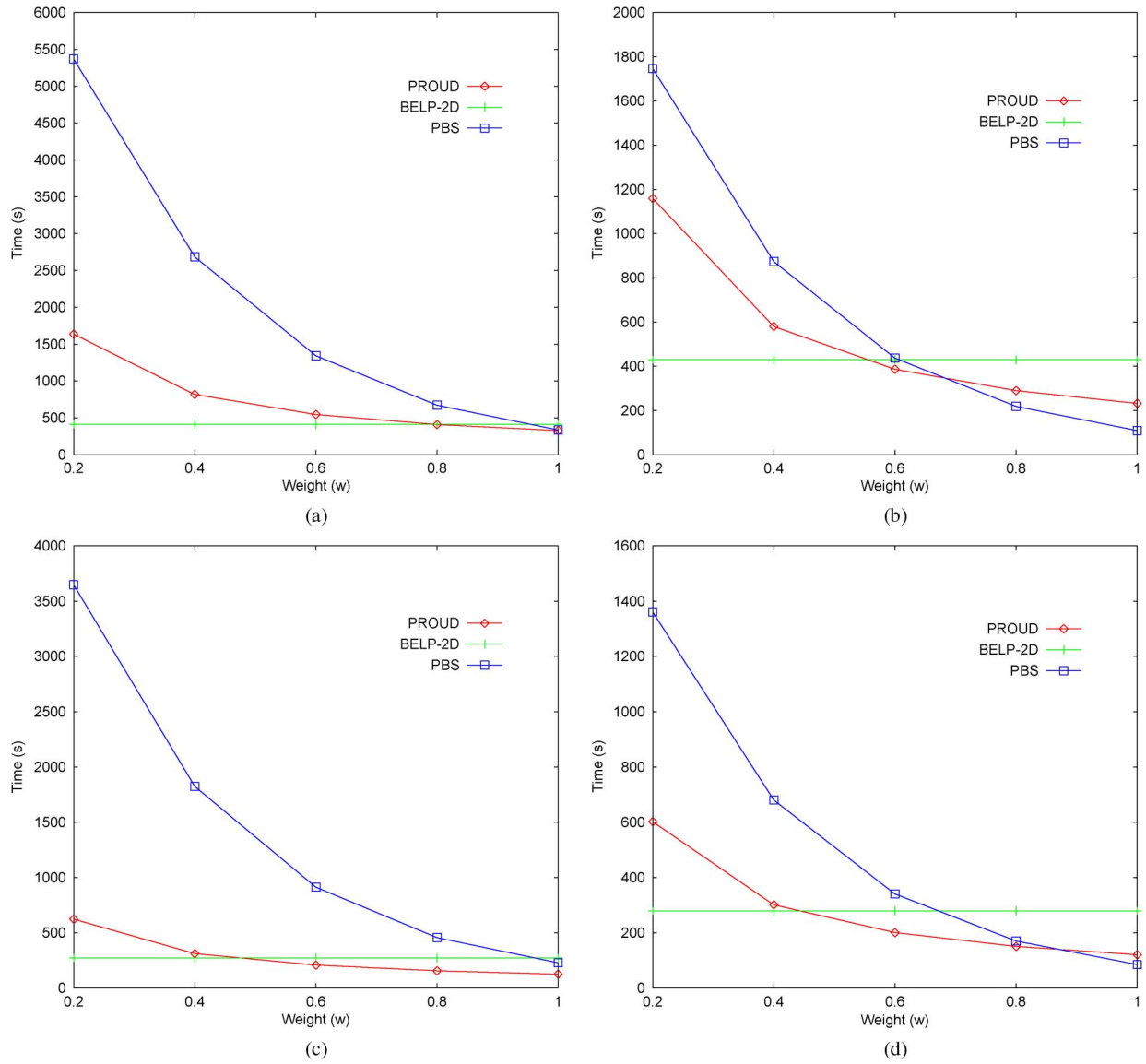

Fig. 5. Actuator interarrival time under (a) uniform random, (b) "Eye" topology, (c) cluster-based uniform, and (d) cluster-based nonuniform sensor distribution.

is also reasonable for sensors with lower weights. This is not surprising given that PBS is customized for the "Eye" topology. Nevertheless, such a topology is not very common in sensor networks.

3) Cluster-Based Uniform Sensor Distribution: We further evaluate our algorithm under cluster-based sensor distribution. Specifically, we place the sensors into three clusters and uniformly and randomly generate the weights of sensors in this experiment.

Similarly, Fig. 5(c) shows the average actuator interarrival time A_{avg} of the three algorithms. Under this cluster-based sensor deployment, PROUD achieves a shorter A_{avg} than both the BELP-2D and PBS algorithms for sensors with high and median weights. PROUD is able to differentiate the sensor visiting frequency and provide the shortest A_{avg} to highly weighted sensors, which satisfies our main objective. An interesting observation is that the A_{avg} under the cluster-based sensor deployment is generally shorter than that under uniform random deployment in all the algorithms. The reason is that the sensors are more concentrated under cluster-based deployment so that they appear to have shorter distances, leading to shorter routes.

4) Cluster-Based Nonuniform Sensor Distribution: We also evaluate our algorithm under a cluster-based nonuniform sensor distribution. Apart from deploying the sensors into three clusters, we also put the sensors with similar weights into one cluster here. The weights of the sensors in the three clusters fall into the ranges 0–0.33, 0.33–0.66, and 0.66–1.0, respectively.

Again, Fig. 5(d) shows the results for the same network with $M = 5$. We observe that PROUD generally performs better than BELP-2D. It achieves a relatively short A_{avg} for sensors with high and median weights. It again differentiates the A_{avg} among sensors according to their weights. PROUD also achieves a comparable A_{avg} with PBS for $w = 1$ and a much lower A_{avg} for all the remaining sensors.

Overall, PROUD performs better than BELP-2D and PBS under various sensor and weight distributions. It generally achieves a shorter A_{avg} than BELP-2D for highly weighted

TABLE II
MINIMUM MOVING SPEED OF ACTUATORS (IN METERS PER SECOND)

Topology	PROUD	BELP-2D	PBS
Uniform random	1.0905	1.3699	3.5805
Eye topology	0.7729	1.4310	1.1649
Cluster-based uniform	0.4157	0.9073	2.4324
Cluster-based non-uniform	0.4013	0.9295	0.9074

TABLE III
ENERGY CONSUMPTION OF ACTUATORS (IN KILOJOULES)

Topology	PROUD	BELP-2D	PBS
Uniform random	30.095	37.538	96.429
Eye topology	21.635	39.167	32.076
Cluster-based uniform	12.118	25.215	65.843
Cluster-based non-uniform	11.735	25.806	25.217

sensors and a much shorter A_{avg} than PBS for most of the sensors.

B. Minimum Moving Speed of Actuators

We next compare the minimum moving speeds of actuators in PROUD, BELP-2D, and PBS. Supposed that the expected average actuator interarrival time o_1 for the sensors with $w = 1$ is 5 min. The corresponding expected average actuator interarrival times for the sensors with weights $w = 0.8, 0.6, 0.4$ and 0.2 are 10, 15, 20, and 25 min, respectively. The minimum moving speeds for the actuators to satisfy these requirements are listed in Table II. The actuators in PROUD can walk at the lowest moving speed among the three algorithms to achieve the above expected average actuator interarrival times in all topologies. BELP-2D requires a higher moving speed than PROUD as its actuator route length is always longer than the average actuator route length in PROUD so that the actuators in BELP-2D have to walk faster than those in PROUD to visit the sensors with $w = 1$ at the same time interval. PBS requires very high minimum moving speeds in both uniform random and cluster-based uniform topologies. This is because its route length is extremely long in these two topologies. Since the sensors with $w = 0.2$ are visited only once in every cycle in PBS, the actuators have to move at a high speed to ensure that the least weighted sensors are visited every 25 min on the average.

To measure the energy consumption of the actuators, we adopt a motion power model [30] based on the popular Pioneer 3DX robots [37]. The motion model is built from real measurement results and is shown to provide very accurate approximation to the actual power consumption [30], [38]–[40].

The motion power $p_m(v)$ (in watts) is given by

$$p_m(v) = 0.29 + 7.5v \tag{13}$$

where v is the moving speed of the actuators in meters per second.

The actuators broadcast their arrivals to the surrounding sensors every 10 s in our simulation. The sensors store their sensing data in the buffer and report them when the actuators approach. Each sensor can buffer up to ten packets for reporting. The energy consumption for the actuators to operate for 1 h is shown in Table III. Clearly, the actuators in PROUD consume less energy than those in BELP-2D and PBS as their moving speeds are lower than those in the other two algorithms. We also find that the energy consumption for communications is usually less than 1 J/h with the above settings, which indicates that the energy consumption for motion constitutes the major part of the total energy consumption.

C. Coordination of Actuators in PROUD

In the previous experiments, we compared the performance of PROUD with BELP-2D and PBS for actuators moving at constant speeds. Different from BELP-2D and PBS, PROUD also considers actuators moving along distinct routes at various speeds. We now investigate the coordination among the actuators with variable speeds in PROUD. We evaluate our algorithm in a network with $M = 8$ and set the expected average actuator interarrival time o_1 for the sensors with the highest weight to be 2 min.

1) Uniform Random Sensor Distribution: Fig. 6(a) shows the minimum moving speeds of actuators under the uniform random sensor deployment. The eight actuators have similar minimum moving speeds as they are walking in the subareas where the sensor locations and weights are randomly generated. It is likely that the actuators will achieve comparable expected route lengths, even when they are walking on different subareas. The figure also shows that the minimum speeds increase with the number of sensors because the actuators need to walk on longer routes to visit more sensors.

2) "Eye" Topology: Fig. 6(b) shows similar results under the "Eye" topology. Since the highly weighted sensors are located only at the center, the expected route lengths here are shorter than that in a network with random sensor distribution. Again, the eight actuators have comparable minimum moving speeds as they are walking in the subareas where the sensor locations are randomly generated, although the sensor weights follow a special eyeball pattern.

3) Cluster-Based Uniform Sensor Distribution: Fig. 6(c) shows the result under the cluster-based uniform sensor distribution. Similar to the previous experiment, the sensors are deployed into three clusters. Again, the weights of sensors are random here. The experiment results show that the required moving speeds of actuators under cluster-based sensor deployment are lower than those under uniform random deployment. The reason is that the sensors are concentrated in smaller areas and therefore can be walked through with shorter routes.

4) Cluster-Based Nonuniform Sensor Distribution: Fig. 6(d) shows the result in a network under cluster-based nonuniform sensor distribution. Similar to the above, three clusters are formed, with the weights falling in the ranges 0–0.33, 0.33–0.66, and 0.66–1.0, respectively.

We observe that the actuator speeds converge to three distinct lines. The effect is particularly obvious in Fig. 6(d), due to its special distribution pattern of sensor weights. The three clusters are consistently walked through by three routes with a constant number of actuators on them. Clusters I, II, and III with weight ranges 0–0.33, 0.33–0.66, and 0.66–1.0 are patrolled by two, three, and three actuators, respectively. Since cluster III has

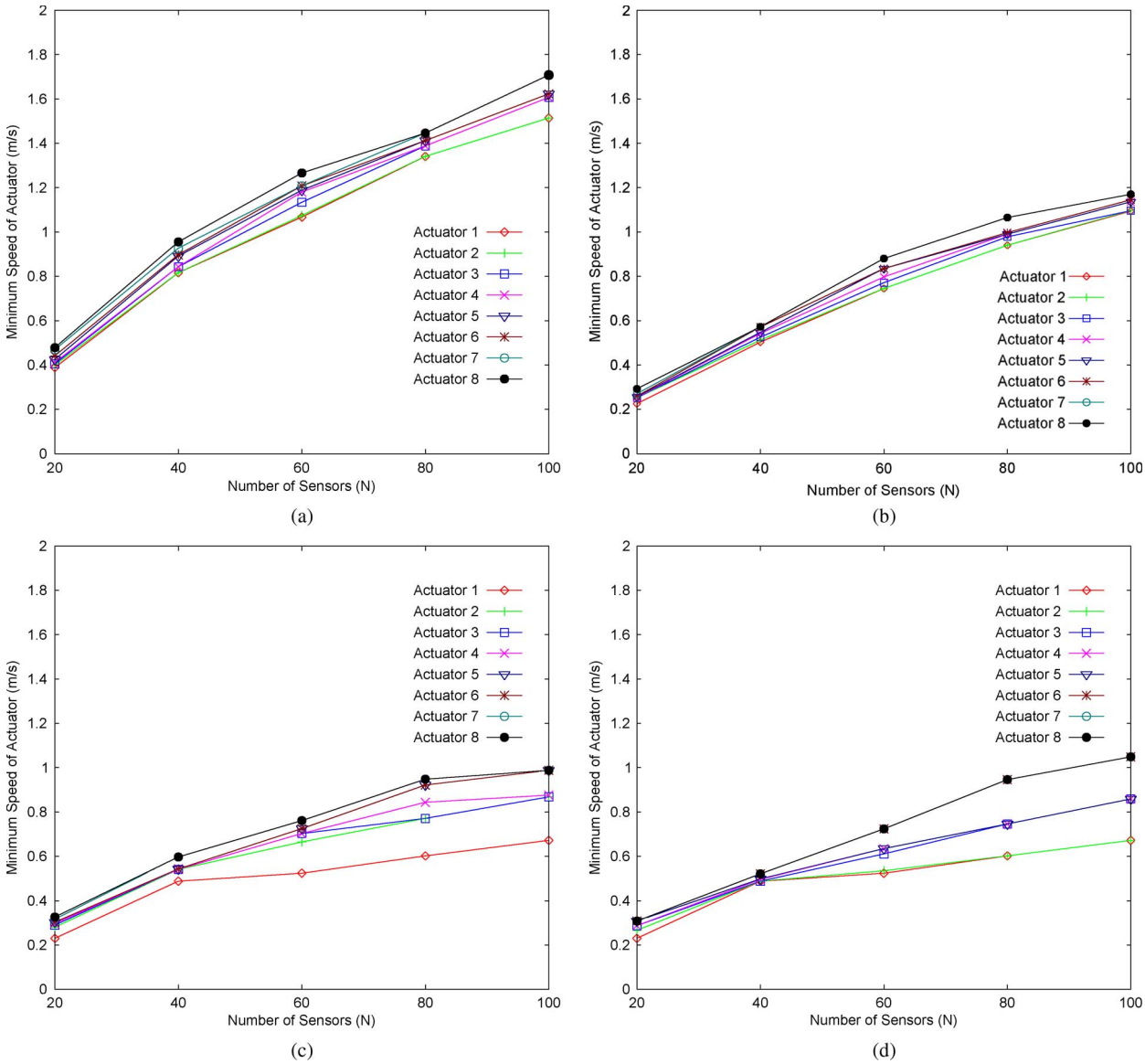

Fig. 6. Minimum speed of actuators under (a) uniform random (b) "Eye" topology, (c) cluster-based uniform, and (d) cluster-based nonuniform sensor distribution.

the longest route length, its actuators (actuators 6–8) require the highest moving speed. On the contrary, the route length in cluster I is relatively short that its actuators (actuators 1 and 2) can walk at the lowest speed.

D. Effectiveness of Multiroute Improvement

We next evaluate the performance of PROUD with the multiroute improvement in this experiment. A network with 100 sensors is deployed with the uniform random distribution, together with two actuators. The actuators are assigned to two subareas at initialization and separately form distinct routes. Since the weights of sensors dynamically change, the two actuators have to accordingly update their routes.

We let the actuators update their routes every 10 min. The speeds of the actuators with and without multiroute improvement are compared. Fig. 7 shows that the two actuators with multiroute improvement walk at closer speeds than those without. It is clear that the multiroute improvement balances the expected lengths of the two routes and effectively reduces the speed difference. Fig. 8 further confirms that the actuators with multiroute improvement can achieve more balanced energy consumption.

E. Effectiveness of Task Exchange Among Actuators

As mentioned earlier, the multiroute improvement may not be applicable in some situations (see Fig. 4), where the task exchange algorithm can instead be applied. We now evaluate task exchange algorithm in terms of the moving speed and energy consumption of actuators. We consider a network with $M = 5$ and $N = 100$ under cluster-based distribution. Again, clusters I, II, and III are formed, which involve sensors with low, medium, and high weights, respectively. The simulation parameters of the energy model are again mainly drawn from [30] and [38].

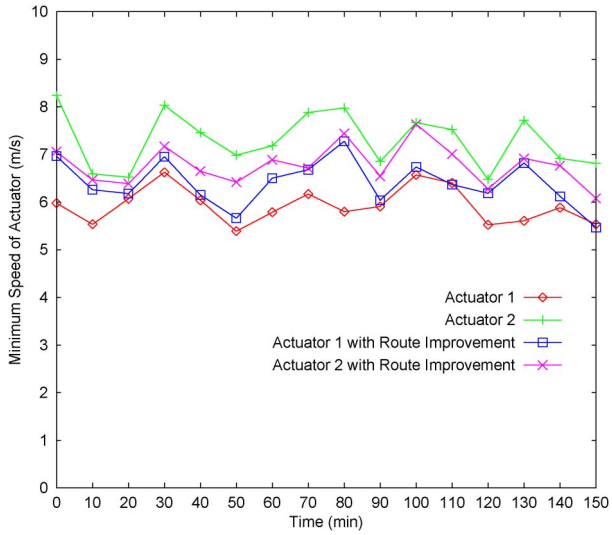

Fig. 7. Speed of actuators with multiroute improvement.

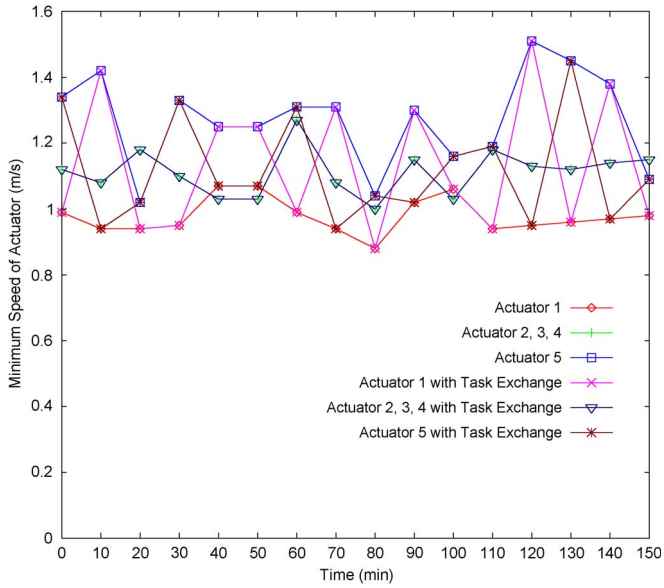

Fig. 9. Speed of actuators with task exchange.

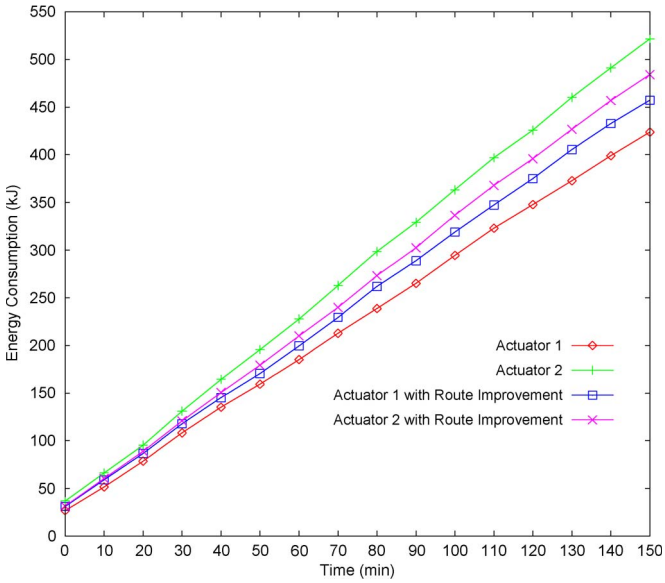

Fig. 8. Energy consumption of actuators with multiroute improvement.

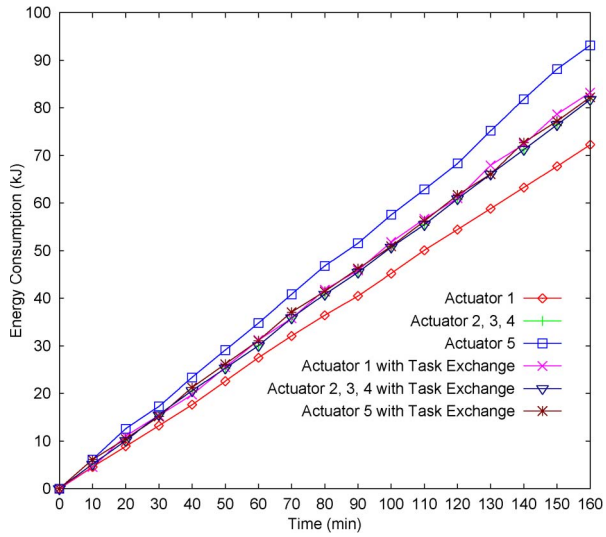

Fig. 10. Energy consumption of actuators with task exchange.

Fig. 9 shows the minimum moving speed of the actuators. Cluster III are assigned with three actuators (actuators 2, 3, and 4) due to the high weights of its sensors, while clusters I and II are both assigned with only one actuator. Since cluster II has a longer expected route length than cluster I, its minimum actuator moving speed keeps higher than the others. As a result, actuator 5 walks at a high minimum speed, while actuator 1 walks at a relatively low speed. This is unfavorable as they have far imbalanced energy consumption, as illustrated in Fig. 10.

With the task exchange algorithm, actuators 1 and 5 interchangeably walk along the routes of clusters I and II to balance their workloads. This can be observed by the two lines of actuators 1 and 5 with task exchange crossing each other in Fig. 9. Eventually, the actuators with task exchange achieve comparable energy consumption, as shown in Fig. 10.

VIII. Conclusion

In this paper, we have focused on WSNs with multiple actuators and their route design. We have proposed an adaptive PROUD algorithm, which aims to minimize the overall interarrival time of actuators with nonuniform sensor weights in a dynamically changing environment. It constitutes a significant departure from traditional static and deterministic mobile element scheduling. In PROUD, the sensors are probabilistically visited by actuators along an *a priori* route. The sensors with higher weights are visited with higher probabilities, enabling shorter actuator interarrival times. Most importantly, the visiting frequencies to sensors can easily be updated by adjusting their visiting probabilities, without frequent route recalculations. We have discussed a distributed implementation of PROUD and extended it to accommodate actuators with variable speeds. We have further proposed a multiroute

improvement and a task-exchange algorithm for evenly distributing the workload among the actuators. Simulation results suggested that PROUD can greatly reduce the average interarrival times in WSANs for highly weighted sensors. It also adapts well to the dynamic change of the network and effectively balances the energy consumption of the actuators.

REFERENCES

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

[2] D. Estrin, R. Govindan, J. Heidemann, and S. Kumar, "Next century challenges: Scalable coordination in sensor networks," in *Proc. ACM MobiCom*, Seattle, WA, 1999, pp. 263–270.

[3] A. Cerpa and D. Estrin, "ASCENT: Adaptive self-configuring sensor networks topologies," *IEEE Trans. Mobile Comput.*, vol. 3, no. 3, pp. 272–282, Jul./Aug. 2004.

[4] W. L. Yeow, C. K. Tham, and W. C. Wong, "Energy efficient multiple target tracking in wireless sensor networks," *IEEE Trans. Veh. Technol.*, vol. 56, no. 2, pp. 918–928, Mar. 2007.

[5] Z. M. Wang, S. Basagni, E. Melachrinoudis, and C. Petrioli, "Exploiting sink mobility for maximizing sensor networks lifetime," in *Proc. 38th HICSS*, 2005, p. 287.1.

[6] J. Luo and J. Hubaux, "Joint mobility and routing for lifetime elongation in wireless sensor networks," in *Proc. 24th IEEE INFOCOM*, Mar. 2005, pp. 1735–1746.

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

[8] E. C.-H. Ngai, Y. Zhou, M. R. Lyu, and J. Liu, "Reliable reporting of delay-sensitive events in wireless sensor-actuator networks," in *Proc. 3rd IEEE MASS*, Vancouver, BC, Canada, Oct. 2006, pp. 101–108.

[9] Y. Gu, D. Bozdag, E. Ekici, F. Ozguner, and C.-G. Lee, "Partitioning-based mobile element scheduling in wireless sensor networks," in *Proc. SECON*, Santa Clara, CA, Sep. 2005, pp. 386–395.

[10] E. C.-H. Ngai, J. Liu, and M. R. Lyu, "An adaptive delay-minimized route design for wireless sensor-actuator networks," in *Proc. 4th IEEE MASS*, Pisa, Italy, Oct. 2007, pp. 1–9.

[11] A. A. Somasundara, A. Ramamoorthy, and M. B. Srivastava, "Mobile element scheduling with dynamic deadlines," *IEEE Trans. Mobile Comput.*, vol. 6, no. 4, pp. 395–410, Apr. 2007.

[12] E. C.-H. Ngai, J. Liu, and M. R. Lyu, "Delay-minimized route design for wireless sensor-actuator networks," in *Proc. IEEE WCNC*, Hong Kong, Mar. 2007, pp. 3675–3680.

[13] P. N. Pathirana, A. V. Savkin, and S. Jha, "Location estimation and trajectory prediction for cellular networks with mobile base stations," *IEEE Trans. Veh. Technol.*, vol. 53, no. 6, pp. 1903–1913, Nov. 2004.

[14] W. Zhao, M. Ammar, and E. Zegura, "A message ferrying approach for data delivery in sparse mobile ad hoc networks," in *Proc. ACM MobiHoc*, Mar. 2005, pp. 187–198.

[15] R. Shah, S. Roy, S. Jain, and W. Brunette, "DATA MULEs: Modeling a three-tier architecture for sparse sensor networks," in *Proc. IEEE Workshop SNPA*, 2003, pp. 30–41.

[16] A. Chakrabarti, A. Sabharwal, and B. Aazhang, "Using predictable observer mobility for power efficient design of sensor networks," in *Proc. 2nd Int. Workshop IPSN*, Apr. 2003, pp. 129–145.

[17] A. Kansal, A. Somasundara, D. Jea, M. Srivastava, and D. Estrin, "Intelligent fluid infrastructure for embedded networks," in *Proc. 2nd ACM MobiSys*, 2004, pp. 111–124.

[18] N. Bisnik, A. Abouzeid, and V. Isler, "Stochastic event capture using mobile sensors subject to a quality metric," in *Proc. ACM MobiCom*, Sep. 2006, pp. 98–109.

[19] Z. Zhang and Z. Fei, "Route design for multiple ferries in delay tolerant networks," in *Proc. IEEE WCNC*, Mar. 2007, pp. 3460–3465.

[20] N. Christofides, A. Mingozzi, and P. Toth, "Exact algorithms for the vehicle routing problem, based on spanning tree and shortest path relaxations," *Math. Program.*, vol. 20, no. 1, pp. 255–282, Dec. 1981.

[21] T. Ralphs, L. Kopman, W. Pulleyblank, and L. Trotter, "On the capacitated vehicle routing problem," *Math. Program.*, vol. 94, no. 2/3, pp. 343–359, Jan. 2003.

[22] L. Lee, K. Tan, K. Ou, and Y. Chew, "Vehicle capacity planning system: A case study on vehicle routing problem with time windows," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 33, no. 2, pp. 169–178, Mar. 2003.

[23] A. Dumitrescu and J. S. B. Mitchell, "Approximation algorithms for TSP with neighborhoods in the plane," in *Proc. SODA*, 2001, pp. 38–46.

[24] J. Bentley, "Fast algorithms for geometric traveling salesman problem," *ORSA J. Comput.*, vol. 4, pp. 387–411, 1992.

[25] S. Arora, "Polynomial time approximation schemes for Euclidean traveling salesman and other geometric problems," *J. ACM*, vol. 45, no. 5, pp. 753–782, Sep. 1998.

[26] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*. Cambridge, MA: MIT Press, 2002.

[27] R. G. Gallager, P. A. Humblet, and P. M. Spira, "A distributed algorithm for minimum-weight spanning trees," *ACM Trans. Program. Lang. Syst.*, vol. 5, no. 1, pp. 66–77, Jan. 1983.

[28] B. Awerbuch, "Optimal distributed algorithms for minimum weight spanning tree, counting, leader election, and related problems," in *Proc. ACM STOC*, 1987, pp. 230–240.

[29] M. Elkin, "A faster distributed protocol for constructing a minimum spanning tree," in *Proc. ACM-SIAM SODA*, 2004, pp. 359–368.

[30] Y. Mei, Y.-H. Lu, Y. C. Hu, and C. G. Lee, "A case study of mobile robot's energy consumption and conservation techniques," in *Proc. IEEE Int. Conf. Adv. Robot.*, 2005, pp. 492–497.

[31] T. He, J. Stankovic, C. Lu, and T. Abdelzaher, "SPEED: A real-time routing protocol for sensor networks," in *Proc. IEEE ICDCS*, Providence, RI, May 2003, pp. 46–55.

[32] E. Felemban, C.-G. Lee, and E. Ekici, "MMSPEED: Multipath multi-SPEED protocol for QoS guarantee of reliability and timeliness in wireless sensor networks," *IEEE Trans. Mobile Comput.*, vol. 5, no. 6, pp. 738–754, Jun. 2006.

[33] TI chipcon, *CC1000 Datasheet*. [Online]. Available: http://focus.ti.com/

[34] Crossbow, *MICA2 Mote Datasheet*. document part no. 6020-0042-08, rev. A. [Online]. Available: http://www.xbow.com/

[35] G. Zhou, T. He, S. Krishnamurthy, and J. A. Stankovic, "Impact of radio irregularity on wireless sensor networks," in *Proc. 2nd Int. Conf. Mobile Syst., Appl., Services MobiSys*, 2004, pp. 125–138.

[36] F. Chen, N. Wang, R. German, AND F. Dressler, "Simulation study of IEEE 802.15.4 LR-WPAN for industrial applications," in *Proc. Wiley WCMC*, 2009, to be published. [Online] Available: DOI: 10.1002/wcm.736

[37] MobileRobots, *Pioneer 3DX Robot Specification*. [Online]. Available: http://www.activrobots.com/

[38] Y. Mei, Y.-H. Lu, Y. C. Hu, and C. G. Lee, "Deployment of mobile robots with energy and timing constraints," *IEEE Trans. Robot.*, vol. 22, no. 3, pp. 507–522, Jun. 2006.

[39] B. P. Gerkey and M. J. Mataric, "Sold!: Auction methods for the multirobot coordination," *IEEE Trans. Robot. Autom.*, vol. 18, no. 5, pp. 758–768, Oct. 2002.

[40] D. Erickson, "Non-learning artificial neural network approach to motion planning for the pioneer robot," in *Proc. IEEE/RSJ Int. Conf. IROS*, Oct. 2003, vol. 1, pp. 112–117.

Edith C.-H. Ngai received the B.Eng., M.Phil., and Ph.D. degrees from the Chinese University of Hong Kong (CUHK), Shatin, Hong Kong, in 2002, 2004, and 2007, respectively.

From 2007 to 2008, she was a Postdoctoral Researcher with the Department of Electrical and Electronic Engineering, Imperial College London, London, U.K. She has conducted research with the VIEW Laboratory, CUHK; the Network Modelling Laboratory, Simon Fraser University, Vancouver, BC, Canada; the Tsinghua National Laboratory for Information Science and Technology, Tsinghua University, Beijing, China; the Intelligent Systems and Networks Group, Imperial College London; and the Networked and Embedded Systems Laboratory, University of California, Los Angeles. She is currently an Assistant Professor with the Department of Information Technology, Uppsala University, Uppsala, Sweden, where she also works with the Uppsala VINN Excellence Center of Wireless Sensor Networks (WISENET). Her research interests include wireless sensor networking, mobile computing, network security, quality-of-service routing, and video information processing.

Jiangchuan Liu (S'01–M'03–SM'08) received the B.Eng. degree (*cum laude*) from Tsinghua University, Beijing, China, in 1999 and the Ph.D. degree from the Hong Kong University of Science and Technology, Hong Kong, in 2003, both in computer science.

From 2003 to 2004, he was an Assistant Professor with the Department of Computer Science and Engineering, Chinese University of Hong Kong, Shatin, Hong Kong. He is currently an Associate Professor with the School of Computing Science, Simon Fraser University, Burnaby, BC, Canada. His research interests include multimedia systems and networks, wireless ad hoc and sensor networks, and peer-to-peer and overlay networks.

Dr. Liu was a recipient of Microsoft Research Fellowship in 2000 and the Hong Kong Young Scientist Award in 2003. He is a co-inventor of one European patent and two U.S. patents. He co-authored the Best Student Paper at IWQoS'08 and the Best Paper from the 2009 IEEE Multimedia Communications Technical Committee. He is a recipient of the Canada NSERC 2009 Discovery Accelerator Supplements Award. He is a member of Sigma Xi. He is an Associate Editor of the IEEE TRANSACTIONS ON MULTIMEDIA and an Editor of the *IEEE Communications Surveys and Tutorials*.

Michael R. Lyu received the B.S. degree from the National Taiwan University, Taipei, Taiwan, in 1981, the M.S. degree from the University of California, Santa Barbara, in 1985, and the Ph.D. degree from University of California, Los Angeles, in 1988.

From 1988 to 1990, he was with the Jet Propulsion Laboratory, Pasadena, CA, as a Technical Staff Member. From 1990 to 1992, he was with the Electrical and Computer Engineering Department, University of Iowa, Iowa City, as an Assistant Professor. From 1992 to 1995, he was a Member of the Technical Staff with the Applied Research Area, Bell Communications Research, Bellcore. From 1995 to 1997, he was a Research Member of the Technical Staff with Bell Laboratories, which was first part of AT&T and later became part of Lucent Technologies. He is currently a Professor with the Department of Computer Science and Engineering, Chinese University of Hong Kong, Shatin, Hong Kong. His research interests include software reliability engineering, distributed systems, fault-tolerant computing, Web technologies, mobile networks, digital video libraries, multimedia processing, and video searching and delivery.

Prof. Lyu has been an Associate Editor for the IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, the IEEE TRANSACTIONS ON RELIABILITY, the *Journal of Information Science and Engineering*, and the *Wiley Software Testing, Verification, and Reliability Journal*. He is an AAAS Fellow and a Croucher Senior Research Fellow.