NETWRAP: An NDN Based Real-Time Wireless Recharging Framework for Wireless Sensor Networks

Cong Wang[1], Ji Li[1], Fan Ye[2], and Yuanyuan Yang[1]

[1] *Department of Electrical and Computer Engineering, Stony Brook University, Stony Brook, NY 11794, USA*
[2] *School of Electronics Engineering and Computer Science, Peking University, Beijing, China*

***Abstract*—Using vehicles equipped with wireless energy transmission technology to recharge sensor nodes over the air is a game-changer for traditional wireless sensor networks. The recharging policy regarding when to recharge which sensor nodes critically impacts the network performance. So far only a few works have studied such recharging policy for the case of using a single vehicle. In this paper, we propose NETWRAP, an NDN based Real Time Wireless Recharging Protocol for dynamic wireless recharging in sensor networks. The real-time recharging framework supports single or multiple mobile vehicles. Employing multiple mobile vehicles provides more scalability and robustness. To efficiently deliver sensor energy status information to vehicles in real-time, we leverage concepts and mechanisms from NDN (*Named Data Networking*) and design energy monitoring and reporting protocols. We derive theoretical results on the energy neutral condition and the minimum number of mobile vehicles required for perpetual network operations. Then we study how to minimize the total traveling cost of vehicles while guaranteeing all the sensor nodes can be recharged before their batteries deplete. We formulate the recharge optimization problem into a Multiple Traveling Salesman Problem with Deadlines (m-TSP with Deadlines), which is NP-hard. To accommodate the dynamic nature of node energy conditions with low overhead, we present an algorithm that selects the node with the minimum weighted sum of traveling time and residual lifetime. Our scheme not only improves network scalability but also ensures the perpetual operation of networks. Extensive simulation results demonstrate the effectiveness and efficiency of the proposed design. The results also validate the correctness of the theoretical analysis and show significant improvements that cut the number of nonfunctional nodes by half compared to the static scheme while maintaining the network overhead at the same level.**

***Index Terms*—Wireless sensor networks, mobile vehicles, named data networking, wireless recharging, mobile energy replenishing, perpetual operation, recharge coordination.**

I. INTRODUCTION

Wireless energy transmission technique [1], [2] has opened up a new dimension to power wireless sensor networks. Compared to opportunistic energy harvesting techniques [3], [4] where the energy supply is not always available, it delivers energy to sensor nodes reliably without the complication of wires or plugs. Mobile charging vehicles (called *SenCar*s [19] in this paper) equipped with charging devices can move around the field and replenish node energy conveniently. The recharging policy - when and which SenCar should recharge which nodes and in what order - critically impacts the efficiency and thus the lifetime of the network.

So far a few works [11], [12] have studied the recharging policy problem. In [11], an optimization problem to maximize the ratio of recharging time to vehicle's idling time is considered. In [12], the recharge problem is studied in a relatively static setting where nodes report their energy levels periodically, and a centralized algorithm computes a specific order so that a single SenCar recharges selected nodes in the next cycle. Then a system-wide optimization is performed to minimize the distance traveled by the SenCar and maximize the network utility. Although commendable first steps, several important practical issues are not considered in the static setting in [11], [12], which may limit its applicability in real network environments.

First, the timely, efficient and scalable gathering of energy status information of nodes and its delivery to mobile vehicles are important and challenging issues in themselves. The previous works do not consider these problems and assume that such information is readily available. Second, it takes nontrivial (e.g., 30-80 min) time to recharge a commercial off-the-shelf battery. Finishing one round of recharging for a network of a few hundred nodes may take several days. During this time the energy levels of nodes may have changed significantly due to unpredictable external events that can trigger extensive activities and quickly drain the battery. The recharging policy computed at the beginning of the cycle is no longer optimal. This can cause energy depletion on some nodes, leading to disruptions to network functions. Third, these algorithms are designed for a single SenCar, which has limited recharging capacity and supports networks of only limited sizes. Multiple SenCars can support larger networks, but new algorithms must be designed to disseminate energy status information to SenCars and coordinate the recharging activities among them efficiently to best leverage their recharging capacities. Finally, the works in [11], [12] use centralized algorithms with high complexity, which may incur extra overhead in computing the optimal solution and may not scale to large network sizes. Because of these practical challenges, a distributed solution is more desirable and efficient in real network environments.

In this paper, we propose a novel real-time energy monitoring and recharging framework that optimizes the recharging policies of SenCars under dynamic network conditions. Instead of letting nodes report their energy levels only after a long period as in [11], [12], a scalable and efficient energy information aggregation protocol gathers battery energy levels continuously from all sensor nodes upon requests by SenCars. The SenCars receive such information and make recharging decisions based on the latest energy information. To deal with unpredictable emergencies where nodes may dramatically drain the battery in short time, the recharging of sensor nodes whose energy levels are below a critical threshold has higher priority and takes precedence over those that can work for relatively longer time with their residual energy.

To ensure high scalability of the proposed framework, we apply *Named Data Networking (NDN)* [14] techniques to gather and deliver energy information to SenCars. To this end, we divide the network in a hierarchical fashion and the energy information is aggregated bottom-up through different levels. NDN uses names instead of locations to address data, which is a natural match for aggregated energy information that belongs to an area instead of any particular node. Thus the aggregated energy information can be addressed by the area's name. NDN also supports mobile vehicles because it has mechanisms to constantly update the routing states in intermediate nodes to follow the movements of vehicles. This is important for the SenCars to receive the energy information timely after moving.

Due to the unpredictable nature of external events, the SenCars may need to deal with multiple concurrent *emergencies* occurring at different locations. How to schedule and coordinate the SenCars to recharge these nodes within their residual lifetimes while minimizing the cost of SenCars is called the *Emergency Recharge Optimization with Multiple SenCars* (EROMS) problem in this paper. We first investigate the necessary conditions for perpetual operations of the network based on energy neutral requirement (i.e., the energy consumed should be less than or equal to the energy replenished) and formally derive the minimum number of SenCars needed to satisfy this condition. Then we show that the EROMS problem can be formulated as a Multiple Traveling Salesmen Problem with Deadlines (*m-TSP with Deadlines*) [35], which is NP-hard. Although heuristic algorithms exist for m-TSP with Deadlines, they are not suitable in our context. First, most of them assume an unlimited number of vehicles while the number of SenCars in our problem is limited. Second, these algorithms consider a relatively static input where the locations to be visited do not change over time. However, in our context, new emergencies may appear and old ones may be resolved as SenCars recharge nodes. Finally, these algorithms may produce unbalanced workloads among SenCars such that some SenCars can be idle while emergencies still exist. Therefore, we propose a new heuristic algorithm to address such deficiencies; it decides which nodes to recharge through a weighted sum of the traveling time and residual lifetime of sensor nodes. We also analyze the complexity of our algorithm and demonstrate its performance to meet dynamic battery deadlines.

We conduct extensive simulations to demonstrate the effectiveness and efficiency of our framework. We trace the process of energy consumption and replenishment, the numbers of emergencies and nonfunctional nodes in different network settings. The results demonstrate that our framework is scalable to networks with hundreds of sensor nodes while ensuring the perpetual operation. Then we compare with the static optimization approach [12] in terms of the fraction of nonfunctional nodes and response time to emergencies. The results show great improvements on reducing the number of nonfunctional nodes and shortening the response time to emergencies. We also validate our theoretical results on energy neutral condition through simulations. Given the network and SenCar parameters, network administrators can easily estimate the minimum number of SenCars needed for perpetual operations.

We make the following contributions in this paper. First, we propose a novel real-time recharging framework for wireless sensor networks, consisting of a set of scalable and efficient NDN-based energy aggregation and gathering protocols. The protocols satisfy both normal and emergency recharging needs for multiple mobile vehicles. Second, we formally analyze the conditions for the minimum number of SenCars needed for perpetual operations. Third, we formulate the emergency recharge optimization with multiple SenCars as an *m-TSP with Deadlines* problem, and further propose an efficient heuristic algorithm suitable to sensor recharging context. Finally, we conduct extensive simulations to demonstrate the effectiveness and efficiency of the framework, compare with existing solutions and validate the correctness of the theoretical analysis. To the best of our knowledge, this is the first effort to apply NDN techniques in sensor networks to make recharge decisions capable of adapting to dynamic network conditions such as emergencies. It is also the first attempt to derive and validate the theoretical minimum number of vehicles needed for perpetual network operation, and the first to formalize the scheduling/coordination of multiple vehicles' recharge activities while meeting sensor battery deadlines and minimizing total traveling cost of vehicles.

The rest of the paper is organized as follows. Section II discusses related work. Section III outlines the framework and assumptions made in the network model. Section IV describes the operations and mechanisms of our protocol followed by Section V and Section VI on deriving the minimum number of SenCars and solving the EROMS problem, respectively. Finally, Section VII presents the simulation results and Section VIII concludes the paper.

II. Related Work

A. Wireless Rechargeable Sensor Networks

Recently, there have much research efforts in wireless energy transmissions from both academia and industry [5], [6], [7], [8], [9], [10], [11], [12], [13]. In [6], the impact of wireless charging technology on sensor networks is investigated using devices from Powercast [5] which are based on radio frequency harvesting, and heuristic algorithms are developed to solve the deployment and routing problem. In [7], deployment problems are studied in a rechargeable sensor network built from an industrial wireless sensing platform and commercial off-the-shelf RFID readers. In [8], an $\mathcal{O}(k^2k!)$ greedy algorithm is designed to find a recharge sequence to maximize the lifetime of sensor nodes using wireless charging, where k is the number of nodes in the recharge sequence. Experimental tests using Powercast devices and insights of wireless energy replenishment are also presented. In [9], a joint routing and wireless charging scheme is proposed to improve network utilization and prolong network lifetime. Implementations based on Powercast devices and typical sensor nodes have been considered.

However, techniques based on radio frequency harvesting are hindered from large deployments due to relatively low efficiency. The seminal works [1], [2] of resonant inductive coupling based wireless energy transmission are capable of transferring a large amount of energy in short time with high efficiency. Several works [10], [11], [12], [13] consider using resonant inductive coupling to recharge sensor batteries. In [10], batteries are allowed to be partially charged and various recharging schemes to traverse the sensing field are explored. In [11], the problem of periodic recharging each sensor node using a single mobile vehicle is considered. A near-optimal solution is provided to calculate the optimal traveling path of the mobile car, by constructing the shortest Hamilton cycle through all sensor nodes. In [12], the problem of jointly optimizing the effective energy recharging and data collection with bounded data latency is studied. A two-step approach is proposed to recharge nodes with the least residual energy while maximizing network utility. In [13], collaboration among multiple vehicles to recharge not only the sensor nodes but also each other vehicles in a line (1-D) network is studied so that a larger network can be covered and vehicles can come back to the starting point.

The above work makes pioneering steps in wireless rechargeable sensor networks. However, several important practical issues are not considered. First, in a practical 2-D network, a single recharging vehicle cannot scale to large network sizes. The recharge coordination problem of dispatching which vehicle to recharge which sensor node so as to minimize the total traveling cost is not studied. Second, how real-time energy information can be aggregated and reported to mobile vehicles efficiently is not considered. Third, the dynamic changes in energy levels that occur inevitably and unpredictably during long recharging cycles are not handled.

B. Named Data Networking (NDN)

Named Data Networking is a new network architecture proposed recently for the Internet [14]. In NDN, data are addressed by their names instead of hosting nodes' locations. The operation is based on two types of messages, *Interest* and *Data*, and the communication is initiated by the receiver. A receiver interested in certain data sends Interest messages carrying the name of the desired data. The Interest message propagates in the network following FIB (Forward Interest Base) states towards nodes hosting desired data. It also leaves a "trail" of PIT (Pending Interest Table) states in intermediate nodes. Once the Interest reaches a node hosting the desired data, Data messages can follow PIT states to traverse back to the receiver.

So far NDN research has largely focused on the Internet, with some efforts on mobile networking. Whether it can be used to satisfy the needs of wireless sensor networks is still unexplored. In this paper, we use wireless recharging as a case study to investigate its applicability in wireless sensor networks. We find that its hierarchical naming structure fits naturally with energy aggregation needs, and its inherent ability to handle mobile receivers is attractive for information delivery to recharging vehicles.

C. Coordination of Mobile Vehicles

Adopting multiple mobile vehicles has been studied for data collection in wireless sensor networks. In [15], a stochastic model is presented to evaluate the performance of data collection when the vehicles follow a symmetric random walk on a 2-D grid topology. In [16], multiple controlled mobile vehicles are adopted for data collection and the objective is to achieve load balancing. In [17], a set of heuristics are proposed to schedule the data collection of multiple mobile vehicles to meet sensors' dynamic buffer overflow time constraints. A sensor may be visited by one or more mobile vehicles depending on its buffer status. In [18], a set of protocols are proposed to achieve spatial coverage equivalence, vehicle mutual avoidance and load balancing. In [20], the problem of minimizing the total traveling cost of multiple mobile vehicles is studied. It formalizes the problem into covering salesman problem and presents a tour-planning heuristic. All the existing work focuses on data collection where mobile vehicles only need to cover sensors in their transmission ranges, and a sensor may be visited by one or more vehicles during a short period. However, in wireless energy replenishment, the effective wireless recharging range is very short compared to data transmissions, while using multiple mobile vehicles to recharge the same sensor node incurs high cost and should be completely avoided.

III. A NOVEL FRAMEWORK FOR WIRELESS RECHARGEABLE SENSOR NETWORKS

In this section, we describe the components, network model and assumptions for our NDN-based wireless recharging framework (NETWRAP). NDN has a few attractive benefits for our environment. First, by sending out new Interest packets, a mobile receiver can continuously update the routing states (i.e., PIT entries) in intermediate nodes. Data can follow the reverse paths traversed by the most recent Interest packets and reach the new location of the receiver. This solves the mobility issue of SenCars and ensures that the latest energy information can reach them in a timely manner. Second, to scale to large network sizes, we divide the network in a hierarchical fashion and energy information is gathered in aggregated forms. Thus the data are bounded to an area rather than any particular node. This makes a natural fit for NDN: the data can be addressed by the area's name. Compared to a flat topology that requires flooding messages throughout the network, such hierarchical aggregation reduces considerable overhead by confining message propagation to parts of the network. Third, NDN provides network robustness when intermediate nodes fail. This is ensured by resending the Interest packet by the receiver when data do not arrive. The new Interest packet explores alternative routes to bypass failures [14].

A. Network Components

The network consists of the following components.

SenCars and Service Station: SenCars query the network for energy information and recharge nodes based on the energy information collected. A service station is used for network management and maintenance. The SenCars can be commanded by the administrator via the service station that has computing and communication capabilities.

Head nodes: A head is a sensor node delegated to aggregate energy information from its subordinate area. When requested by a SenCar or by the head of its upper level, a head queries energy information from subordinate sub-areas at the lower level, aggregates such information and sends to the requester.

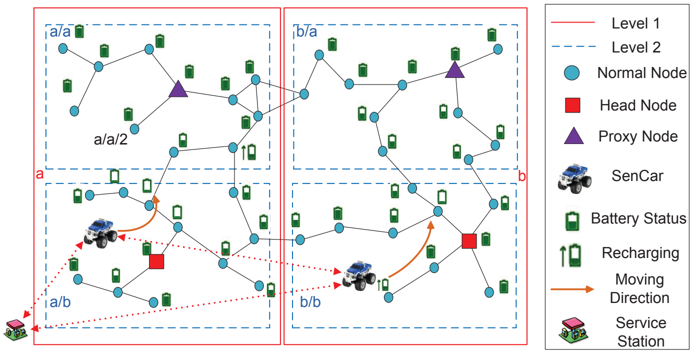


Fig. 1. Illustration of area names and network components.

The head node can be changed dynamically depending on its energy.

Proxy: An *emergency* occurs when a node's residual energy falls below a pre-determined emergency threshold (e.g., 10%). Such events need to be taken care of immediately. A proxy node aggregates emergencies from sensor nodes and reports such information to the SenCar when queried. Only top-level head nodes are proxies.

Normal Node: A sensor node not selected as a head is a normal node. It reports energy information to head nodes, or sends emergency directly to its proxy when its energy level drops below an emergency recharge threshold.

B. Name Assignments and Network Model

We assume sensors are scattered uniformly randomly in the network. The network is divided into several areas and each area is further divided recursively. The division of the area is based on geographical coordinates of the sensing field. Each division generates some new *sub-areas* and increases the number of *levels* in the network. This process repeats until the bottom level subarea becomes small enough such that during the time a SenCar recharges such a subarea, the normal energy levels of the network do not change too much to warrant interruption of that recharge. Fig. 1 gives an example of a 2-level network with 2 areas (solid lines), each further split into 2 sub-areas (dashed lines), where each sub-area on the second level contains about 10 sensor nodes.

Based on the results of area divisions, we assign NDN data names for different subareas in a hierarchical manner. For example, Fig. 1 shows all the name assignments for different subareas (e.g., the first level areas are "a" and "b", and the second level has "a/a", "a/b", ...). Thus each subarea is identified by its unique hierarchical name. Each node has an ID including the name of its containing bottom-level subarea plus an identifier. For example, "a/a/2" is node "2" in subarea "a/a". The hierarchical names make it easy to confine the propagation scope of a message to any subarea: nodes beyond the intended subarea (carried by the message) can simply stop further propagation.

In addition, we have the following assumptions: 1) Sensor nodes are stationary and each node knows its location. 2) Nodes have the same transmission range and messages are forwarded over multiple hops in large networks. 3) The SenCars have positioning systems and know their locations. The IDs and locations of all sensor nodes, and the subarea names are known to the SenCars (e.g., through a one-time effort during network initialization). 4) The SenCars are equipped with powerful antennas so that they can communicate among themselves and to the service station directly using long range communication technologies (e.g., cellular, WiMax). 5) Sensors might perform different tasks thus the energy consumption is not uniform among nodes. 6) When a SenCar finds that its own energy is about to deplete, it moves back to the service station for battery replacement. Since the traveling and battery replacing time is short compared to recharge time, we ignore the time for SenCar's battery replacement. 7) The SenCar can approach sensors in close proximity and induce enough currents on sensor's receiving coils for battery recharge.

IV. THE NDN-BASED REAL-TIME WIRELESS RECHARGING PROTOCOL

In this section, we present the detailed design of NETWRAP. We first give an overview of the protocol design in Section IV-A. Then we describe different operating phases of the protocol in Section IV-B.

A. Protocol Overview

In NETWRAP, the SenCars obtain the most up-to-date energy information from sensors and makes recharge decisions in real time. The energy information is aggregated on head nodes at different levels. To be robust, the head is usually selected as the node having the maximum energy level in its area. This selection process is done at the beginning of network startup through the propagation of *head selection* messages. The details will be discussed in the next subsection.

To start energy information collection, SenCars send out *energy interest* messages to poll the *heads* on the top-level. Once the heads receive such messages, they send lower level *energy interest* messages to their child-heads in respective subordinate areas. This process is repeated down the head node hierarchy, until finally the bottom-level *energy interest* messages reach the nodes in the bottom-level subareas.

Once a sensor node receives a bottom-level *energy interest*, it responds by sending out an *energy* message containing its ID and battery level. When the heads on the bottom-level receive such *energy* messages, they select sensor nodes with energy level below a normal recharge threshold (defined by the administrator), and send the names of these nodes and their energy information in an *energy* message to their parent head nodes. This is repeated up the head node hierarchy, until finally the top-level head nodes have the aggregated energy information and send it to the SenCar. When multiple SenCars query energy information simultaneously, the top-level head nodes send the aggregated energy information to the nearest SenCar. Thus SenCars recharge nearby normal nodes to reduce their travel costs. The details are explained later in this section.

To reduce transmission overhead, the head is delegated partial responsibilities to pre-select sensor nodes to be recharged. At the bottom level, this is done by letting heads select nodes with low energy levels. At upper levels, a head selects the subordinate area which can be recharged with the most amount of energy. Thus the SenCars can replenish the network with more energy in one movement.

Such normal energy aggregation is conducted at the requests of the SenCars. For emergency nodes that have dangerously low battery levels below an emergency threshold, they send out *emergency* messages to the proxy that manages its area. The route to its proxy is built by *head selection* messages from the proxy.

After completing the recharge of any node, a SenCar sends out an *emergency interest* message to query whether any emergency has appeared. These messages are directed to proxies where lists of emergency nodes are stored. The proxies respond by sending back the emergency node IDs, estimated residual lifetimes and energy levels. SenCars receive the messages and use the emergency recharge algorithm to decide which nodes to recharge. Note that different from normal recharging, the SenCar recharging an emergency node may not be the nearest one. This is due to the urgency to avoid any battery depletion. When a head node is low on energy, it can choose another node with high energy, and send out a *head notification* message to notify the latter to become the new head.

B. Protocol Design

We describe the detailed protocol assuming the head hierarchy has l levels.

1) Head Selection: After the areas and names have been configured, the network performs head selection from the bottom up starting at the l-th (i.e., the lowest) level. Since initially sensor nodes have about the same level of energy, any of them may become a head. Each sensor node i generates a random number x. If $x > K$, where K is a pre-determined threshold, the node floods a *head selection* message in its l-th level subarea, containing the name of this subarea, $x_{max} = x$, and ID_{max} set to its own ID. Otherwise the node waits for messages from other nodes in the area.

A node receiving such a *head selection* message compares the x_{max} in the message with its local record x_{max}. If its local record is larger, the message is discarded. Otherwise, the sensor updates its local x_{max} to that in the message, sets ID_{max} to that in the message, and forwards the message to its neighbors except the node that sent this message. Finally, the node with the maximum x wins the election and is recorded by all the nodes in this subarea as the head.

New heads at the l-th level then contend to become heads of the $(l-1)$-th level. They flood new *head selection* messages in the $(l-1)$-th level subarea, carrying the area's name, their respective x values and IDs. Intermediate nodes perform similar comparisons. This will elect the heads at the $(l-1)$-th level. This process is repeated recursively until head nodes of all levels are elected.

One difference for the *head selection* messages starting from the $(l-1)$-th level and up is that messages carrying smaller x than the local copy are not discarded. Instead, they are propagated throughout the respective subarea. This builds routing states in intermediate nodes of the subarea: An intermediate node has one entry for each child-head, pointing to the neighbor from which the message from that head arrives first. Duplicate copies of the same message arriving later are discarded.

Such states are effectively FIB (Forward Interest Base) entries in NDN. Later a parent head at the $(k-1)$th-level can send *energy interest* messages to its child-heads at the k-th level using such states. To build FIB entries for the 1st level

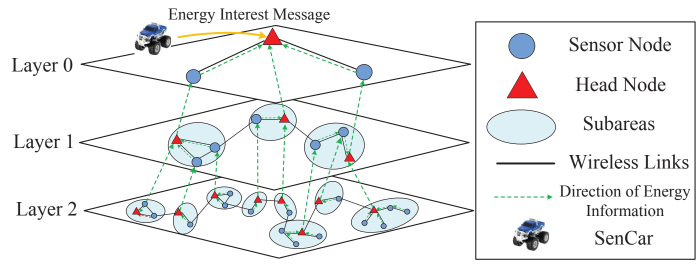


Fig. 2. Example normal energy information convergence.

head nodes, they each flood a top-level *head selection* message throughout the whole network. Later the energy interest queries from the SenCars can use such states to reach them.

2) Normal Energy Interest Propagation: After the head hierarchy is constructed, the SenCars send *energy interest* messages to query for nodes needing recharge. The energy information is gathered on demand, and top down in the hierarchy. We will describe normal energy information collection first. Emergency information is collected similarly, but with only top-level heads involved to reduce latency.

For normal energy information, interest messages are sent by the SenCars (e.g., with data name set to "/energy/normal/a" to collect energy information from area a). Intermediate nodes use the FIB entries established by top-level *head selection* messages to forward it to all top-level head nodes. To guide the return of future data from a top-level area, an intermediate node also sets up a PIT (Pending Interest Table) entry pointing to the neighbor from which the interest message towards this area is received. Fig. 2 gives a pictorial illustration of a network with 3 levels. After an interest message is sent by the SenCar, the energy information is converged from the bottom level to the top level towards the SenCars.

To avoid duplicate selection of the same normal nodes and reduce travel costs, we want only the nearest SenCar to receive a head node's normal energy information. To this end, the *energy interest* message from each SenCar carries a hop count increased by one at each intermediate node. When multiple such messages towards the same top-level area are received, an intermediate node updates its PIT entry to record only the neighbor sending the message with the smallest hop count. Later energy information from a head can follow such directions to reach the nearest SenCar. After an intermediate node forwards energy information from a top-level area, it deletes the corresponding PIT entry.

Upon receiving an energy interest message, a top-level head sends a new energy interest message to its child-heads, with the data name set to all subareas of its children (e.g., from the head node of area /a, "/energy/normal/a/*"). Similarly, these messages reach all child-heads following FIB entries. Intermediate nodes also set up PIT entries so that later energy information from child-heads can go back to their parent head. This process is repeated down the hierarchy, until finally heads at bottom-level flood their respective subareas with interest messages.

3) Normal Energy Report and Node Recharge: When a sensor node receives an l-th level *energy interest* message, it responds with an *energy* message including its ID and residual energy. With the help of PIT entries, the message is returned to the head of the l-th level.

The head examines if the reported residual energy is less than the normal recharge threshold. If so, the ID of the node is added to a list, and the energy that can be recharged to this node is added to a summation counter. After the head has collected these messages, it sends an aggregation message, containing the list, the summation counter and its subarea name to its parent head. A parent head compares such messages from its child-heads, selects the one with the largest summation counter (i.e., the bottom-level subarea that can be recharged of the greatest amount of energy), and forwards to its parent head. This process is repeated upwards in the hierarchy. Finally, the SenCar nearest to a top-level head receives a message for the bottom-level subarea with the largest summation counter. It moves there and recharges those nodes in the ID list one by one. Only after recharging those nodes will the SenCar sends another normal energy query.

The reason we delegate selection partially to head nodes is twofold. First, we expect much less variation in normal energy levels. Thus the SenCar can choose one bottom-level subarea and finish recharging all listed nodes. Only after the whole subarea is recharged, we expect enough changes in normal energy distribution that warrants a new normal energy query from the SenCar. Second, this also keeps the return message sizes small and reduces overhead.

4) Emergency Energy Report and Node Recharge: Emergency energy report is slightly different due to the urgency. If a node detects that its energy level is below the emergency recharge threshold, it immediately sends an *emergency* message containing its ID and energy level to its proxy (i.e., its top-level head node). Because the head node floods a top-level *head selection* message during head election, the same FIB entries can be used to forward emergency messages to the head.

Instead of waiting for recharging a whole bottom-level subarea, a SenCar sends out *emergency interest* messages to each proxy after finishing recharging any single normal node or emergency node (e.g., with data name set to "/energy/emergency/a" to collect emergency information from the proxy of area a). To guide the return of future data from the proxy of a top-level area, an intermediate node sets up a PIT entry pointing to the neighbors from which the emergency interest messages towards this area are received. Later emergency information from a proxy can follow such directions to return to the SenCar. After an intermediate node forwards emergency information from a proxy, it deletes the corresponding PIT entry.

When an *emergency interest* message is received, the proxy returns its list of IDs, energy levels and estimated residual lifetime of emergency nodes, if there exists any. Compared to normal recharge that maintains nodes' energy at medium levels so that they do not deplete energy very soon, emergency recharge requires more accurate information on how long a node can last, i.e., an estimation of the residual lifetime. In this paper, it is calculated by taking a weighted sum of the estimated residual lifetime in previous time slots. The estimated lifetime L_i at time slot i is obtained by dividing the current energy consumption rate from residual energy. L_i can be obtained by a weighted average of previous lifetime at n time slots

$$L_i = \frac{L_{i-n} + L_{i-n+1}}{2^n} + \frac{L_{i-n+2}}{2^{n-1}} + \cdots + \frac{L_i}{2}.$$

For simplicity, we restrict the previous lifetime n used in the calculation to 3. For example, the estimated residual lifetime in time slot t_0, t_1, t_2 and t_3 is calculated as L_0, $L_0/2 + L_1/2$, $L_0/4 + L_1/4 + L_2/2$, and $L_0/8 + L_1/8 + L_2/4 + L_3/2$, respectively. We can see that our estimation gives a dominated weight of lifetime in the current slot, which is similar to estimating process time in operating systems [37].

The SenCar uses the algorithm in Section II to decide which node to recharge. It switches back to normal operation mode only when no emergency is reported. When multiple SenCars query emergency information simultaneously, they coordinate with each other and make an optimal decision to assign the emergency nodes to each of them. The procedure of emergency assignment is described in Section II.

5) Head Hierarchy Maintenance: A head can be short on energy, which can happen once in a while because the head usually engages in more activities than a normal node. When this happens, a new head is needed. Because only heads of bottom-levels contend for higher level elections, a head at any level is always the head of its bottom-level subarea. It receives the energy reports from normal nodes in its bottom-level subarea upon the normal interest query from the SenCar. So it can choose a node with the highest energy, and floods a *head notification* message to notify all nodes in the bottom-level subarea of the new head.

The new head then triggers a new head election process in its $(l-1)$-th level subarea. It propagates a new *head selection* message in its $(l-1)$th subarea, but carrying its energy level instead of the random number x. Other heads in this $(l-1)$-th level subarea do the same. Then a new $(l-1)$-th head with the maximum energy is elected. If this is the same head, the process stops. Otherwise, the new $(l-1)$-th level head triggers the same process in its upper level subarea, until finally a new top-level head is elected. Note that although our head selection and maintenance scheme shares some similarities with previous studies [21], in our scheme, NDN techniques are adopted to provide scalable communication between head nodes of different levels, which is new compared to previous work.

V. THEORETICAL ANALYSIS OF ENERGY NEUTRALITY AND MINIMUM NUMBER OF SENCARS

In this section, we study a couple of important theoretical questions. Given a sensor network, what is the necessary condition for it to operate perpetually and what is the minimum number of SenCars needed to satisfy this condition? For a rechargeable sensor network to operate perpetually, the *energy neutral* condition must be satisfied, i.e., for each sensor node the energy replenished is no less than the energy consumed in any arbitrarily long time period.

We use a simple Bernoulli process to model a node's energy consumption: with probability p it consumes unit energy in a unit time slot [23]. The assumption is quite natural for most of the sensing applications. For example, in an event-based

TABLE I
TABLE OF NOTATIONS

Notation	Definition
$\mathcal{N}$	Set of sensor nodes with N elements
$\mathcal{M}$	Set of emergency nodes with M elements at the time when SenCar makes a query
$\mathcal{S}$	Set of SenCars with S elements
N	Number of sensors in the network
p	Probability a node consumes unit energy in a time slot
n	Number of time slots
R_n	Energy replenished for a node in n time slots
E_n	Energy consumed for a node in n time slots
E_0	Initial energy of a sensor node
C	Total battery capacity
t_r	Maximum recharge time of a sensor node
t_i	Recharge time of node i
α	Weight parameter $\alpha \in [0, 1]$
$\mathcal{L}$	Set of residual lifetime of emergency nodes

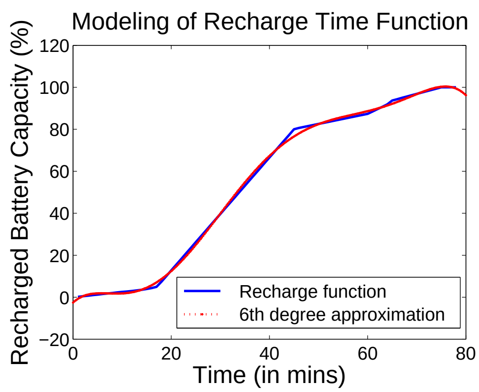


Fig. 3. Recharge function and approximation function by Matlab curve fitting.

sensor network, events occur sporadically and are governed by a Poisson distribution. Once the unit time slot is small enough such that only one event can occur during a unit time slot, the Poisson distribution is equivalent to Bernoulli distribution [24]. A long time period consists of n unit time slots. Let R_n and E_n denote the energy replenished and consumed for a sensor node in n time slots, respectively, and E_0 denote the node's initial energy. Table I summarizes the general notations and their corresponding definitions in this paper.

In principle, wireless recharge is limited by how fast the coil can transfer energy (induction between two coils) and how fast the battery can absorb energy. The lesser of the two determines the recharging rate. We assume that the transmitting coil can induce enough recharge currents on the receiving coil. Resonant inductive coupling based wireless energy transfer can easily achieve this. It has been shown [1] that it can transfer 60 watts of power in excess of 2 meters. When the recharge current is high enough, the recharge time depends on how fast the battery can absorb the charge, which is a battery characteristic. To model the relation between battery capacity and recharge time, we use an example of Panasonic Ni-MH AAA batteries with 780 mAh based on available curves from the data sheets [22]. The instantaneous voltage at time instance t is mapped to capacity ratio and the resultant recharge time function is shown in Fig. 3. Then we use curve fitting in Matlab to obtain an approximation function. [1]

[1]The 6-th degree polynomial equation gives a closed approximation to the original recharge time function $c(t) = -2.7872\times10^{-8}t^6+6.814\times10^{-6}t^5-6.138\times10^{-4}t^4+0.02405t^3-0.3541t^2+2.12t-2.526$ where t is the recharge time.

The energy neutral condition is

$$R_n + E_0 \geq E_n \tag{1}$$

Eq. (1) states that for a long time period n, on each sensor node, the sum of replenished energy and initial energy should be at least as large as the consumed energy. This is a *necessary condition* for the perpetual operation of the network.

From Eq. (1), we can derive the minimum number of SenCars needed, S. We first estimate a loose upper bound for R_n in terms of S. Since it is analytically intractable to obtain R_n, an upper bound can provide reasonable estimates of the total recharge capabilities from the SenCars. Intuitively, SenCars reach their maximum recharging capacity when they can "barely" keep up with the recharging needs, which means that they keep recharging node after node without any idle time in between, and each node has almost zero energy before being recharged. A SenCar can replenish at most the battery's full capacity in the full recharge time[2]. During n time slots, the total recharged energy for the whole network is the recharge rate, $(C/t_r)S$, times the time duration n (e.g., battery capacity C = 780 mAh, full recharge time t_r = 73.4 min for a Panasonic Ni-MH AAA battery [22]). The recharged energy R_n is averaged on each sensor node by dividing the number of sensor nodes N in the network. Thus, the upper bound of R_n is $\frac{nCS}{t_rN}$.

Note that on the right hand side of Eq. (1), E_n is a random variable. We have the following lemma.

Lemma 1. *The probability for the energy neutral condition to hold is* $P_{op} = \Phi\left(\frac{R_n+E_0-np}{\sqrt{np(1-p)}}\right)$.

Proof: Let $X_1, X_2, \ldots, X_n$ be independent and identically distributed Bernoulli random variables for energy consumption in each time slot with probability p. $E_n = \sum_{i=1}^{n} X_i = n\overline{X}$. When n is sampled over a long time period, by the Central Limit Theorem, we know $\overline{X} \sim (p, \frac{p(1-p)}{n})$. Thus, E_n is also normally distributed with $\mu(n) = np$ and variance $\sigma^2(n) = np(1-p)$ $(E_n \sim \mathcal{N}(np, np(1-p)))$ [25]. Hence,

$$P_{op} = Pr\{R_n + E_0 > E_n\} = \Phi\left(\frac{R_n + E_0 - \mu(n)}{\sqrt{\sigma^2(n)}}\right)$$

and the energy neutral condition holds with P_{op}. ■

Proposition 1. *The minimum number of SenCars required to achieve perpetual operation is*

$$S = \left\lceil \frac{t_rN(2.33\sqrt{np(1-p)} + np - E_0)}{Cn} \right\rceil$$

Proof: Since $\Phi^{-1}(1) \to \infty$, we consider the sensor network achieves perpetual operation when $P_{op} \geq 0.99$. In other words, to guarantee the sensor network to operate perpetually, the probability that the energy replenished is larger than the energy consumed is 0.99. From $\Phi^{-1}(0.99) \leq \frac{\frac{nCS}{t_rN}+E_0-np}{\sqrt{np(1-p)}}$, we obtain the minimum number of SenCar S. ■

[2]Different from [10], where partial recharge is allowed, we assume fully recharging batteries to avoid "memory effects" that can reduce the number of charge cycles, so that battery lifetime can be maximized.

To illustrate how Proposition 1 can be used to estimate the minimum number of SenCars, we consider a concrete example comprised of 500 sensor nodes with the recharge function of Panasonic Ni-MH AAA batteries and calculate the minimum number of SenCars needed. Assume in an application the battery can last for 5 days on average without recharge and the energy consumption follows the Bernoulli distribution. It yields $\mu(1) = 37.5mJ$ and $\sigma^2(1) = 9.4mJ^2$ for a time period of 6 months in which each time slot is 1 second. Using Proposition 1, we can calculate $S = \lceil 2.41 \rceil = 3$ which means it needs at least 3 SenCars to cover 500 nodes under the given energy consumption rate. We can see that the derivation from Proposition 1 can help the network administrator plan the network. Once the experimental parameters and the application specifics from the sensors have been determined (e.g., network size N, recharge time t_r, initial energy E_0, working probability p, operation duration n and battery capacity C), we can easily obtain the minimum number of SenCars needed. As will be seen later, the correctness of the derivation is also verified in simulations.

VI. EMERGENCY RECHARGE OPTIMIZATION ALGORITHM

In this section, we study the Emergency Recharge Optimization with Multiple SenCars problem (EROMS). Our objective is to minimize the total traveling cost of the SenCars while guaranteeing recharge before sensors' battery depletion. We formalize this problem into a Multiple Traveling Salesmen Problem with Deadlines. We show the problem is NP-hard and propose a heuristic algorithm suitable for dynamic real-time recharging.

A. Problem Formulation

Given a set of SenCars $\mathcal{S}$ and a set of emergency nodes $\mathcal{M}$, we formalize the problem as follows. Consider a graph $G = (V, E)$, where $V_0^{(k)}$ is the starting position of SenCar k, and V_i ($i \in \mathcal{M}$) is the location of emergency sensor i to be visited. E is the set of edges. Each edge E_{ij} has a latency cost $c_{ij} = t_i + t_{ij}$, where t_i is the time to recharge node i from its current energy level to full capacity, and t_{ij} is the traveling time from node i to node j. For SenCar k, $c_{0j}^{(k)}$ represents its cost from its initial position 0 to node j. For each sensor node i, the residual lifetime is L_i. A_i specifies the arrival time for a SenCar at sensor node i.

We introduce decision variables x_{ij} for edge E_{ij}. The decision variable is 1 if an edge is visited, otherwise it is 0. Additionally, $x_{0j}^{(k)}$ is 1 if SenCar k moves from its initial position to node j. u_i is the position of vertex i in the path. We virtually make the SenCars return to $V_0^{(k)}$ after recharging all the selected nodes by setting $c_{i0}^{(k)} = 0, i \in \mathcal{M}$, thus the EROMS problem can be formulated as the Multiple Start Traveling Salesman Problem with Deadlines in which multiple traveling salesmen start from different locations to visit a set of cities within their deadlines.

$$\mathbf{P1}: \quad \min \left\{ \sum_{i=1}^{M} \sum_{j=1}^{M} c_{ij} x_{ij} + \sum_{k=1}^{S} \sum_{j=1}^{M} c_{0j}^{(k)} x_{0j}^{(k)} \right\} \tag{2}$$

Subject to

$$\sum_{j=1}^{M} x_{0j}^{(k)} = \sum_{i=1}^{M} x_{i0}^{(k)} = 1, \forall k = 1, 2, \ldots, S, \tag{3}$$

$$\sum_{i=1}^{M} x_{ik} = \sum_{j=1}^{M} x_{kj} = 1; \forall k = 2, \ldots, M, \tag{4}$$

$$A_i \leq L_i; \forall i = 1, 2, \ldots, M, \tag{5}$$

$$x_{ij} \in \{0, 1\}; \forall i, j = 1, 2, \ldots, M, \tag{6}$$

$$2 \leq u_i \leq M; \forall i = 2, 3, \ldots, M, \tag{7}$$

$$u_i - u_j + (M - S) x_{ij} \leq M - S - 1; \quad \forall i, j = 2, 3, \ldots, M, i \neq j. \tag{8}$$

Constraint (3) guarantees that the recharge path starts at 0 and finishes at 0. Constraint (4) ensures the connectivity of the path and that every vertex is visited at most once. Constraint (5) guarantees that the arrival time of the SenCar is within sensor's residual lifetime. Constraint (6) imposes x_{ij} to be 0-1 valued. Constraints (7) and (8) eliminate the subtour in the planned route. The subtour elimination constraints are formulated according to [26], [27].

We now show that EROMS is NP-hard. The classic Multiple Traveling Salesman Problem (m-TSP) can be considered as a special case of the EROMS problem with the deadline to visit a city extended from 0 to infinity, and all the SenCars starting from one position. A solution to the EROMS problem would give a solution to m-TSP. Since m-TSP is known to be NP-hard [28], the EROMS problem is NP-hard.

B. Minimum Weighted Sum Heuristic Algorithm and Complexity Analysis

In this subsection, we propose a heuristic algorithm for the EROMS problem that jointly considers the residual lifetime and traveling time. In general, m-TSP is closely related to the Vehicle Routing Problem (VRP) in which a fleet of vehicles start from the same depot and visit client locations except that in m-TSP, salesmen are allowed to start from different locations. The m-TSP with Deadlines can be considered as a special case of the *Multiple Traveling Salesman Problem with Time Windows* (m-TSPTW)[3]. This problem is similar to Vehicle Routing Problem with Time Window (VRPTW) which has been studied in the literature and a handful of optimal and approximation algorithms are available [29], [30], [31], [32], [33], [34], [35], [36].

The approaches to VRPTW are usually divided into two phases. A construction of a feasible tour is sought in the first phase and the tour is interactively improved in the second phase. In [29], several heuristic algorithms are proposed and a computational evaluation is presented to study the performance. In [30], a local search algorithm is proposed to reduce the computation of checking feasibility constraint of TSPTW. In [31], the minimum number of vehicles to meet the time window requirements is studied by utilizing precedence graphs. An insertion heuristic to TSPTW is proposed in [36]. The scheme builds a route by inserting a node each time. While checking the time feasibility, backtracking is sometimes needed and the

[3] m-TSP with Deadlines is m-TSPTW having all release time at 0.

solution is further improved in the post-optimization phase. However, since checking the tour feasibility is as hard as the original problem [30], these approaches are still quite computationally expensive.

On the other hand, several approximation algorithms have been proposed for the VRPTW problem in [32], [35]. However, these algorithms are not suitable for the recharging problem context. First, they assume the number of vehicles is unlimited but the number of SenCars is bounded in our problem[4]. Second, existing algorithms deal with a static problem input. However, in EROMS, new emergencies may appear at any time, and residual node lifetimes also vary due to ongoing sensing activities. Maintaining an optimal schedule would become prohibitively expensive. Finally, existing algorithms may generate unbalanced workloads among SenCars, resulting in idling SenCars while emergencies still exist.

Thus we present a heuristic algorithm that schedules recharge assignments among SenCars. Two important metrics affect the recharging order between node i and node j: the traveling time between node i to node j, and their residual lifetime L_i and L_j. If node j has a small L_j such that it would be dead if a SenCar recharges node i first, node j should be visited first.

We use a weighted sum w_{ij} of traveling time from the current node i to next node j and the residual lifetime of node j,

$$w_{ij} = \alpha t_{ij} + (1-\alpha) L_j. \tag{9}$$

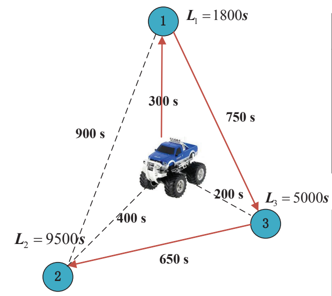
w_{ij} is used to decide which node j to recharge next. A sensor node with a smaller weighted value should be visited at a higher priority. When $\alpha = 1$, the algorithm reduces to nearest node selection that the SenCars always recharge the closest node first regardless of battery deadlines; when $\alpha = 0$, it picks the node with the earliest battery deadline first regardless of the traveling distance.

The value of α greatly affects the schedule. Fig. 4 gives an example of a SenCar and 3 sensor nodes. The residual lifetime and the traveling time on each edge are shown in the figure, and α varies from 0, 0.5 to 1. We assume that recharging a sensor battery takes 1 hour to finish. At time 0 s, the SenCar calculates the weight to sensor nodes 1, 2 and 3. The minimum weights are circled. When $\alpha = 1$, node 3 has the minimum weight; when $\alpha = 0.5, 1$, node 1 has the minimum weight. However, if node 3 is chosen to be the next node, node 1 would have been dead after finishing recharging node 3. Thus $\alpha = 1$ is infeasible in this example. $\alpha = 0.5$ and $\alpha = 0$ generate the same schedule $1 \to 3 \to 2$.

From this example, we can see that the value α affects the feasibility of the solution. We might expect that the total distance be inversely proportional to α and a binary search may locate the maximum feasible α. However, some tests have shown that it is not always true. We decide to search through a list of A distinct α values, e.g., $\alpha = 0.0, 0.1, \ldots, 1.0$ where $A = 11$. We find that this choice achieves a desirable tradeoff between optimality and complexity.

When a SenCar performs calculation, it communicates via a long range radio with other SenCars to know their positions

[4]Adopting limited number of vehicles usually requires a relaxed time constraint which allows late arrivals [33], [34]. However, the time constraint should not be relaxed in the EROMS problem.



Current time: 0s

Weight	$\alpha = 1$	$\alpha = 0.5$	$\alpha = 0$
w_{01}	300	1050	1800
w_{02}	400	2600	9500
w_{03}	200	4950	5000

Current time: 3900s

Weight	$\alpha = 0.5$	$\alpha = 0$
w_{12}	3250	5600
w_{13}	925	1100

Fig. 4. Example of Minimum Weighted Sum Algorithm.

TABLE II
MINIMUM WEIGHTED SUM ALGORITHM

```
Input: weight parameter α ∈ [0,1] in stepsize 1/(A − 1),
position of SenCar at node k, emergency set M, traveling
time from i to j, t_ij, residual lifetime L_i, ∀i, j ∈ M,
node list Ω_i at service station, i ∈ N
Output: result weight parameter α and schedule sequence Q
Initialize minDist = ∞
For α = 0, ..., 1
 While M ≠ ∅
  compute weight w_kj ← α t_kj + (1 − α) L_j
   communicate service station If
   Ω_i = 1, Set w_ki = ∞
   End if
   find j ← arg min_j w_kj
   Q_t ← Q_t + j, M ← M − j
   update ∀i ∈ M, L_i ← L_i − t_kj − t_j
   If L_i ≤ 0
   declare infeasible and break (Inform service station).
   End if
   move to position j, k ← j, recharge and update L_j
  End while
  If feasible
  compute total cost dist(Q_t)
   If dist(Q_t) < minDist,
   minDist ← dist(Q_t), Q ← Q_t
   End if
  End if
End for
```

for computing the weighted sum. Since the SenCar is equipped with high-capacity batteries and powerful antennas, using long range communication among the SenCars and service station is a reasonable option here. Current technology such as cellular communications and WiMax can support data transmission up to several miles, which would suffice the scale for a sensor network. To avoid conflicts where multiple SenCars choose the same node for recharge, we utilize the service station to store and update the availability of each node. The procedure is similar to that for shared memory access in operating systems [37]. The service station maintains a 0-1 valued node list Ω. Once a sensor is chosen, its value is set to 1 (locked). Otherwise, it is 0. The value should be changed back from 1 to 0 when a SenCar finishes recharging that node. A SenCar can simply communicate with the service station, exclude nodes already selected by other SenCars, and notify the service station of the status of nodes it chooses. Table II shows the pseudo-code of the entire algorithm.

We now analyze the complexity of the heuristic algorithm. Note that the node selection operations are executed on each SenCar, which takes $\mathcal{O}(M)$ time. For each SenCar, it performs

TABLE III
PARAMETER SETTINGS

Parameter	Value
Field Length	200×200, $282 \times 282m^2$
Number of Nodes N	250, 350, 500, 1000
Number of SenCars S	1, 2, 3, 4, 5
Number of Levels	3
Areas on l-th level	4^l
Battery Capacity	780 mAh
Transmission Range	18 m
Unit Energy Consumption r_c	37.5 mJ
Energy Consumption Probability p	0.5
SenCar Speed	1 m/s
Maximum Recharge Time	73.4 mins
Normal Recharge Threshold	50%
Emergency Recharge Threshold	10%
Simulation Time	6 months

M/S rounds of node selections and the total number of tests on α is A. Thus, the total computational complexity of our heuristic algorithm is $\mathcal{O}(\frac{AM^2}{S})$.

VII. PERFORMANCE EVALUATION

In this section, we use simulation to evaluate the effectiveness and efficiency of our framework. We have developed a discrete event-driven simulator using POSIX thread programming in C language. Message communications between sensor nodes are emulated using inter-process communication in our simulator. Our simulator is fully capable of realizing message communication based on NDN (message routing), information convergence, recharge and SenCar mobility. To evaluate the performance of NETWRAP, we examine two network sizes of 500 and 1000 sensor nodes, uniformly randomly distributed over a $200 \times 200m^2$ and $282 \times 282m^2$ square field, respectively. The field size is chosen so that the two cases have the same node density. The network consists of 3-level hierarchy with 4^l number of subareas at the l-th level. The energy consumption on each sensor is a Bernoulli random variable with probability p to consume unit energy (37.5 mJ). If a sensor node works continuously at this rate, the battery can last for 5 days.

To evaluate energy overhead of the protocol, we use the model presented in [38], i.e., $e_t = (e_1 d_r^\alpha + e_0)l$, where e_t is the energy consumption while transmitting a message of l bits, d_r is the transmission range, e_1 is the loss coefficient per bit, α is the path loss exponent and e_0 is the excessive energy consumed on sensing, coding, modulations, etc.[5] The relationship between recharged energy and recharge time follows that of Panasonic Ni-MH AAA battery [22]. To understand the impact of the number of SenCars on network performance, we show marginal cases where the number of SenCars is not sufficient while adding one more SenCar would guarantee perpetual operations. These cases are $S = 2, 3$ for $N = 500$ and $S = 4, 5$ for $N = 1000$. We will present these cases in the following and validate the correctness of Proposition 1. All the parameter settings in the simulation are listed in Table III.

A. Evaluation of Weighted-sum Algorithm

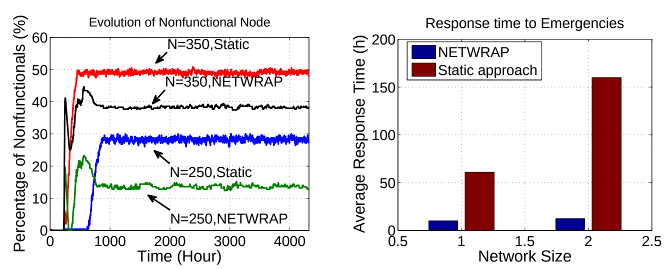
In this subsection, we evaluate the effectiveness of the weighted-sum algorithm in finding the shortest path and achieving no node failure. We examine cases when 4 SenCars are employed. We assume the locations of emergencies are randomly distributed in the field of $282 \times 282m^2$, and the residual energy uniformly distributed from zero to the emergency threshold. The corresponding residual lifetime is calculated by dividing the residual energy by pr_c, the expected energy consumption in unit time.

[5] $d_r = 15$m, $e_0 = 45 \times 10^{-9}$ J/bit, $e_1 = 10 \times 10^{-9}$ J/bit, $\alpha = 2$.

TABLE IV
TOTAL TRAVELING DISTANCE OF SENCARS, D

M	D ($\alpha = 0$)	D ($\alpha = 0.2$)	D ($\alpha = 0.4$)
72	7524.1	7473.3	7740.2
80	7652.4	7578.9	7706.6
88	8662.6	8128.3	7251.6
96	Infeasible	Infeasible	Infeasible
M	D ($\alpha = 0.6$)	D ($\alpha = 0.8$)	D ($\alpha = 1$)
72	6843.5	**6390.6**	Infeasible
80	7271.8	**6941.0**	Infeasible
88	**6998.3**	Infeasible	Infeasible
96	Infeasible	Infeasible	Infeasible

Table IV shows the total distance of SenCars when the number of concurrent emergencies M increases from 72 to 96 in a step of 8. Note that when the number reaches 96, the set of 4 SenCar is not sufficient to resolve all the emergencies without complete battery depletion. For $M = 88$, weight parameter $\alpha = 0.8, 1$ are not feasible and for $M = 72, 88$, $\alpha = 1$ is not feasible either. We notice that in the case when $\alpha = 1$, some nodes that suffer from energy shortage may not get recharged in a higher priority thereby rendering the result infeasible to avoid battery depletion. As we can see from this example, the choice of α is critical, when α approaches 1, the total distance is decreased at the risk of becoming infeasible. Thus we need to search for α in our algorithm. In real applications, the value of α is subject to change and determined by real-time statistical data and parameters.



(a) (b)

Fig. 5. Comparison of static and real-time approaches in terms of (a) Percentage of nonfunctional nodes; (b) Average response time to emergencies.

B. Performance Comparison with a Static Optimization Approach

In this subsection, we compare network performance of our real-time framework with the static optimization approach used in [12]. Since the static approach has only designed algorithms for a single SenCar, we set the number of SenCar at 1 and compare the percentage of nonfunctional nodes and response time to emergencies when $N = 250, 300$. The percentage of nonfunctional nodes indicates how many nodes have depleted energy and are waiting for the SenCar. The response time to emergencies is measured from the time a node reports emergency until it is resolved by the SenCar. A shorter response time indicates that the SenCar can respond faster to emergencies.

In the static approach, the SenCar selects nodes with energy less than the normal recharge threshold, calculates the

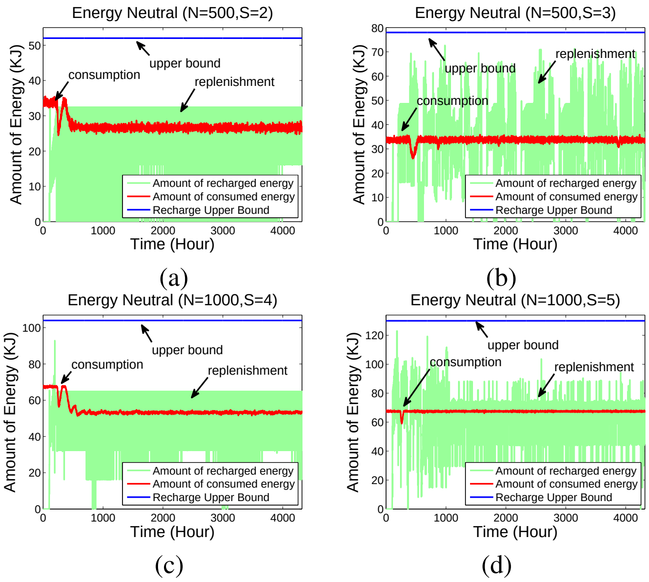


Fig. 6. Evolution of energy consumption vs. energy replenishment in 6 months time. (a) $N = 500$, $S = 2$. (b) $N = 500$, $S = 3$. (c) $N = 1000$, $S = 4$. (d) $N = 1000$, $S = 5$.

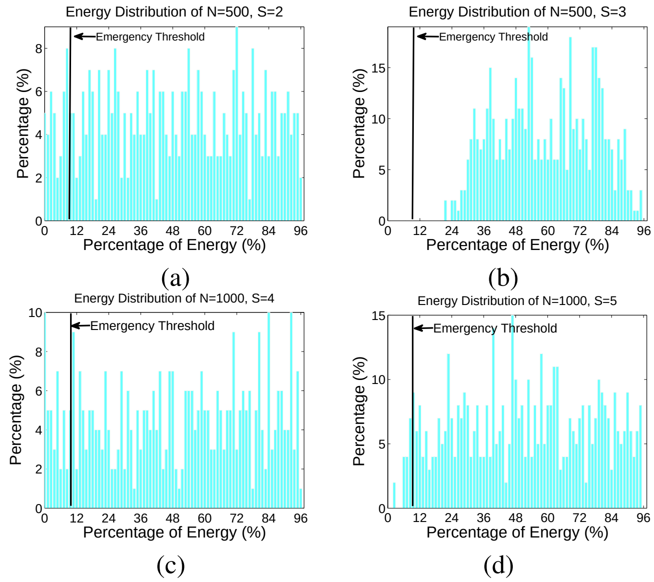


Fig. 7. Energy distribution at equilibrium. (a)$N = 500$, $S = 2$. (b) $N = 500$, $S = 3$. (c) $N = 1000$, $S = 4$. (d) $N = 1000$, $S = 5$.

minimum traveling distance throughout these nodes and performs recharge one by one. Fig. 5(a) shows the percentage of nonfunctional nodes. We can see the number of nonfunctional nodes is much higher in the static optimization approach, e.g. when $N = 250$, there are around 15% nonfunctional nodes in NETWRAP but nearly 30% in the static approach. This is because that some nodes in the pre-computed sequence may consume energy at faster rates, making the initial sequence computed in the static method no longer valid. Thus it cannot cover all the sensor nodes before energy depletion. Second, a node in emergency is not treated with priority in the static method. Thus a node in emergency may deplete its energy before the SenCar arrives, resulting in high percentage of nonfunctional nodes. The results in Fig. 5(a) clearly indicate that our real-time framework is more effective in recharging sensor nodes and resolving emergencies.

Fig. 5(b) shows the average response time to emergencies. We can see while NETWRAP takes around 10 and 12.5 hours (for $N = 250$ and 350 respectively), the static approach takes drastically longer times (around 61 and 160 hours, almost one order of magnitude longer). This is because in [12] emergency and normal nodes are not differentiated. A pre-computed route containing both types of nodes would result in extremely long waiting times for emergency nodes. The approach degrades fast and becomes infeasible as the network size increases. In contrast, NETWRAP prioritizes nodes in emergency; it resolves nonfunctional situations much faster than the static approach.

C. Network Performance

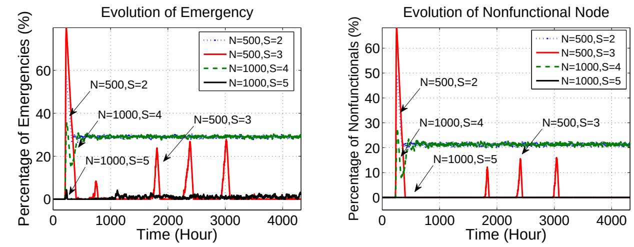
In this subsection, we evaluate the energy evolution of the network, the number of emergency and nonfunctional (i.e., energy depleted) nodes of the network, and the maintenance cost of the framework.

1) Energy Evolution: First, we show the energy evolution in the networks with 500 nodes and 1000 nodes served by different numbers of SenCars, compared to the upper bounds of total recharge capabilities. In Fig. 3, the maximum recharging rate is achieved in the 17-th minute to 45-th minute duration. It replenishes 75% of the battery energy, equivalent to 433 J/min. The upper bound is calculated by assuming that all the SenCars are performing recharge at maximum recharging rates all the time. In Fig. 6, the amount of energy consumed, replenished and recharge upper bound in every one-hour time slot is plotted as functions of the simulation time.

In Fig. 6(a) and (c), we can see that the consumed energy "steps down" to a lower level around 400 hours and then enters equilibrium. This is because a portion of sensor nodes deplete their energy and do not get recharged. In these two scenarios the energy neutral condition has been violated, simply because the number of SenCars is not enough. Fig. 6(b) and (d) show the energy evolution when the number of SenCars is increased by 1, both of which satisfy the energy neutral condition at the equilibrium and there is no such "step-down" effect in energy consumption. The gap between the recharge upper bound and energy replenished is due to that there are traveling time and idling time between two consecutive recharges in simulations. In addition, the SenCars may perform normal recharge in which the recharging rates are much lower than the maximum recharging rates used for calculating the upper bound.

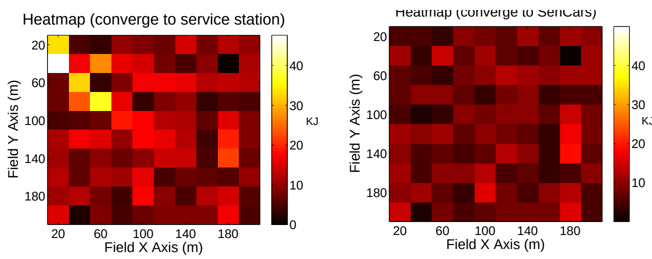
The energy distribution among nodes also carries valuable information about the health of the network. Higher average energy distribution is more robust to unexpected surges in energy consumption. Fig. 7 shows the energy distribution of $N = 500$, $S = 2, 3$ and $N = 800$, $S = 4, 5$. To see the benefits of more SenCars, compare Fig. 7(a) to Fig. 7(b). The latter has energy distribution that concentrates around a higher average value. In Fig. 7(d) for a network size of 1000 sensors, the number of nodes with energy below the emergency threshold is significantly lower than that in Fig. 7(c).

2) Number of Emergencies: Fig. 8 compares the percentage of nodes in emergency and nonfunctional (i.e., energy at zero) status for networks of 500 and 1000 nodes with different numbers of SenCars. First, we can see that there are surges in the numbers of emergency and nonfunctional nodes during the first 200 hours. This is due to the fact that the SenCars only responds to requests when the node energy is below the normal



(a) (b)

Fig. 8. Number of emergent and nonfunctional nodes (a) number of emergent nodes (b) number of nonfunctional nodes.



(a) (b)

Fig. 9. Comparison of energy information convergence schemes (a) converge to service station (b) converge to SenCars.

recharge threshold. When such requests swarm into the job queues on the SenCars at the beginning of 200 hours, we can see that the SenCars' capacity has been temporarily exceeded. As the energy of sensors is restored, the numbers of emergency and nonfunctional nodes decrease sharply.

To illustrate the consequences of insufficient number of SenCars, we vary the number of SenCars S over a range including the minimum number needed for energy neutral. Fig. 8 (a) and (b) show the number of emergency and nonfunctional nodes over time. For cases $N = 500, S = 2$ and $N = 1000, S = 4$ when the number of SenCars is insufficient for energy neutral, we can see that about 30% nodes are in constant emergency and 20% nodes are in nonfunctional status after the network achieves equilibrium. For $N = 500, S = 3$, there are occasional nonfunctional nodes but they are soon recharged by the SenCars. For a majority of the time, the number of nonfunctional nodes stays at zero. For $N = 1000, S = 5$, the number of nonfunctional nodes stays at zero at equilibrium with only a small number of emergencies.

Recall from Proposition 1 that the minimum number of SenCars for $N = 500$ and $N = 1000$ can be calculated as $S = \lceil 2.41 \rceil = 3$ and $S = \lceil 4.84 \rceil = 5$ for the given parameter settings in Table III. These numbers match well with our simulation results that $S = 3, 5$ are the minimum number of SenCars to achieve perpetual operation at equilibrium, respectively. By utilizing our theoretical analysis, the network administrator can make reasonable estimations for the minimum number of SenCars needed when planning a network.

D. Cost Evaluation

1) Comparison of Energy Information Collection Schemes: We compare energy consumption for different energy information collection schemes. Rather than collecting energy information at the SenCars, another method is to route it through multi-hop transmission to the service station. The service station computes recharge schedules and disseminates decisions to SenCars via long range radio. A challenge to this alternative

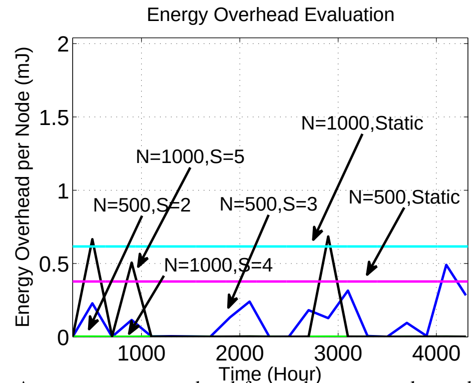


Fig. 10. Average energy overhead for each sensor node per hour.

scheme is that more energy is consumed on nodes near the service station. For demonstration purposes, we draw the heat map of energy consumed in a one-hour interval after the network enters equilibrium in Fig. 9. First, we observe that more energy is consumed if the information is routed back to the service station, i.e., 3-4 times of that to route it to SenCars. Second, more energy is consumed on nodes near the service station, which is shown as bright spot in Fig. 9(a). Since a rechargeable battery has a limited number of recharging cycles, higher loads on these nodes result in more frequent recharge and faster battery expiration.

2) Evaluation of Protocol Overhead: We evaluate the energy overhead incurred during transmission of all types of messages sent by sensor nodes or SenCars, and compare with that of the static scheme in [12], where energy information is routed to the service station every 6 hours. Fig. 10 shows the average energy overhead for each node per hour in a 6 month period for different scenarios. Due to head selection, our protocol has certain amount of overhead (around 6-8 mJ/h per node) at the beginning. We plot the energy overhead after the networks enter equilibrium.

First, we can see that the average energy overhead is from 0.1 to 1 mJ. Compared to the average energy consumption of 135 J/h (i.e., average energy consumption per slot 37.5 mJ times 3600s) due to sensing activities for each node, the overhead is negligible. Second, we can see that the average energy overhead is similar to that the static scheme. Intuitively, our protocol could incur more energy overhead because energy information is collected more frequently. However, the static scheme requires the energy information routed back to the service station. Thus unbalanced energy consumption on nodes near the service station is inevitable. The scalability of the scheme degrades when the size of the network increases. In contrast, delivering the energy information to multiple mobile SenCars alleviates the unbalance in our protocol.

The energy overhead to gather emergency energy information is not significant in our protocol. Both the emergency interests from SenCars and emergency reports from nodes are sent directly to proxies (i.e., top-level heads) without propagating through the hierarchy, leading to less energy overhead. More importantly, when the number of SenCars is sufficient, most of the time the network has very small fraction of nodes in emergency.

Normal energy information gathers causes more overhead. Upon receiving *energy interests*, the heads need to poll their descendants in a top-down manner which finally results in the

TABLE V
BALANCE OF LOAD ON SENCARS

SenCar	1	2	3	4	5
$N = 500, S = 2$	51%	49%	-	-	-
$N = 500, S = 3$	35%	34%	31%	-	-
$N = 1000, S = 4$	25%	25%	25%	25%	-
$N = 1000, S = 5$	22%	20%	20%	19%	19%

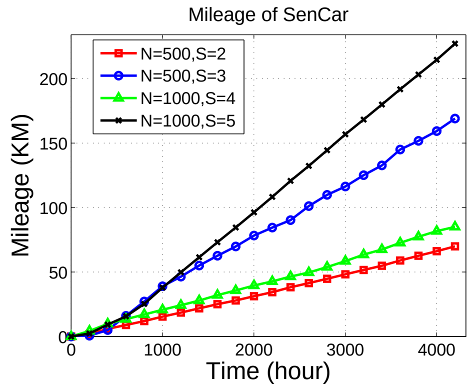


Fig. 11. Mileages SenCars have traveled in 6 months.

broadcast of *energy interest* message in subareas at the bottom-level. Such broadcast, as well as the reply from each node, leads to the increase of the number of messages transmitted and the overhead. When the number of SenCars is sufficient as $N = 500, S = 3$ and $N = 1000, S = 5$, more energy overhead is observed.

3) Evaluation of Load Balance and Mileage on SenCars: We monitor the energy replenished by each SenCar and compare their workloads. The workload is measured by the amount of energy replenished to sensor nodes during the entire simulation period. Table V shows that the workloads are well balanced in all four scenarios due to the effective coordination in our framework. The SenCars share the work evenly and no SenCar is overloaded. On the other hand, we use the mileages SenCars travel to evaluate the cost (e.g., the energy consumed) for SenCars to move around. Fig. 11 shows the accumulated mileages in 6 months. For both network sizes, the networks with fewer SenCars (500 nodes and 2 SenCars, 1000 nodes and 4 SenCars) have lower mileage compared with the same network with more SenCars (500 nodes and 3 SenCars, 1000 nodes and 5 SenCars), respectively. This is due to the presence of nonfunctional nodes. According to the calculation of weight for emergency selection (Eq. (9)), decision is made based on the residual lifetime of the nodes and the traveling time from the SenCars to the nodes. For the networks with fewer SenCars, there are always approximate 20% nonfunctional nodes after the networks enter equilibrium. The weights are dominated by the traveling time which is proportional to the distances from the SenCars to these nodes. Thus the SenCars always choose the nearest nodes for recharge. For the network with more SenCars, however, the traveling time is not the dominating factor, thus the SenCars may choose a farther node with shorter residual lifetime for recharge to avoid battery depletion. This causes the increase of SenCar mileage.

VIII. CONCLUSION AND FUTURE WORKS

In this paper, we study how to coordinate multiple mobile vehicles to recharge sensors and propose a novel NDN-based real-time wireless energy replenishment framework. In the framework, a comprehensive set of communication protocols are developed using NDN concepts and mechanisms to provide real-time energy information gathering and delivery. The protocols can adapt to unpredictable network conditions and satisfy the needs for both normal and emergency recharging. Then we formally analyze the probability for the energy neutral condition to hold, which is required for perpetual network operation. In the analysis, we derive the minimum number of SenCars needed to achieve perpetual operations. To address concurrent emergency situations, we model the Emergency Recharge Optimization problem with multiple SenCars into the m-TSP with Deadline problem and provide a fast, efficient heuristic algorithm suitable for dynamic network conditions. The extensive simulation results demonstrate the efficiency and effectiveness of the proposed algorithm and the framework in improving network performance compared to the static optimization approach and other methods of energy information collection. We also validate the correctness of theoretical analysis on the minimum number of SenCars needed.

There are some interesting issues that deserve further study in future. One important practical aspect is the recharge capacity of the SenCar, and its own energy consumption in moving. Such factors should be included in the optimization formulation to understand their impacts on recharge decisions. In addition, since the resources are limited on the SenCars, important nodes (e.g., cluster heads, proxies) may need some priority in recharging. Another important issue is the distributed decision and coordination among SenCars. Currently, the decisions are aided by the service station with long range radio transmissions. A distributed design would allow SenCars to coordinate with each other to make decisions, which is more robust and potentially more efficient.

ACKNOWLEDGMENTS

This work was supported in part by the US NSF grant numbers ECCS-0801438 and ECCS-1307576 and US ARO grant number W911NF-09-1-0154.

REFERENCES

[1] A. Kurs, A. Karalis, R. Moffatt, J. D. Joannopoulos, P. Fisher and M. Soljacic, "Wireless power transfer via strongly coupled magnetic resonances," *Science*, vol. 317, pp. 83, 2007.

[2] A. Karalis, J. D. Joannopoulos and M. Soljacic, "Efficient wireless non-radiative mid-range energy transfer," *Annals of Physics*, vol. 323, no. 1, pp. 34-48, Jan. 2008.

[3] M. Rahimi, H. Shah, G. Sukhatme, J. Heideman and D. Estrin, "Studying the feasibility of energy harvesting in a mobile sensor network," *IEEE International Conference on Robotics and Automation*, 2003.

[4] A. Kansal, J. Hsu, M. Srivastava and V. Raqhunathan, "Harvesting aware power management for sensor networks," *43rd ACM/IEEE Design Automation Conference*, 2006.

[5] PowerCast Corp, "http://www.powercastco.com."

[6] B. Tong, Z. Li, G. Wang and W. Zhang, "How wireless power charging technology affects sensor network deployment and routing," *IEEE ICDCS*, 2010.

[7] S. He, J. Chen, F. Jiang, D. Yau , G. Xing and Y. Sun, "Energy provisioning in wireless rechargeable sensor networks," *IEEE INFOCOM*, 2011.

[8] Y. Peng, Z. Li, W. Zhang and D. Qiao, "Prolonging sensor network lifetime through wireless charging," *IEEE RTSS*, 2010.

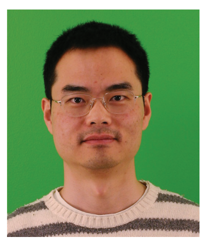
[9] Z. Li, Y. Peng, W. Zhang, and D. Qiao, "J-RoC: a Joint Routing and Charging Scheme to Prolong Sensor Network Lifetime", *IEEE ICNP*, 2011.

[10] C. Angelopoulos, S. Nikoletseas, T. Raptis, C. Raptopoulos, F. Vasilakis,"Efficient energy management in wireless rechargeable sensor networks," *IEEE MSWiM*, 2012.

[11] Y. Shi, L. Xie, T. Hou and H. Sherali, “On renewable sensor networks with wireless energy transfer,” *IEEE INFOCOM*, 2011.
[12] M. Zhao, J. Li and Y. Yang, “Joint mobile energy replenishment and data gathering in wireless rechargeable sensor networks,” *IEEE ITC*, 2011.
[13] S. Zhang, J. Wu, and S. Lu, “Collaborative Mobile Charging for Sensor Networks,”*IEEE MASS*, 2012.
[14] V. Jacobson, D. Smetters, J. Thornton, M. Plass, N. Briggs and R. Braynard, “Networking named content,” *ACM CoNEXT*, 2009.
[15] R. Shah, S. Roy, S. Jain and W. Brunette, “Data MULEs: Modelling a three-tier architecture for sparse sensor networks”, in *Proc. of the First IEEE International Workshop on Sensor Network Protocols and Applications*, 2003.
[16] D. Jea, A. Somasundara and M. Srivastava, “Multiple controlled mobile elements (data mules) for data collection in sensor networks”, in *IEEE DCOSS*, 2005.
[17] A. Somasundara, A. Ramamoorthy and M. Srivastava, “Mobile element scheduling for efficient data collection in wireless sensor networks with dynamic deadlines,” *IEEE RTSS*, 2004.
[18] I. Chatzigiannakis, A. Kinalis, S. Nikoletseas and J. Rolim, “Fast and energy efficient sensor data collection by multiple mobile sinks,” *ACM MobiWac*, 2007.
[19] M. Ma and Y. Yang, “SenCar: an energy-efficient data gathering mechanism for large-scale multihop sensor networks,” *IEEE Trans. Parallel and Distributed Systems*, Vol. 18, No. 10, pp. 1476-1488, 2007.
[20] M. Ma and Y. Yang, “Data gathering in wireless sensor networks with mobile collectors,”*IEEE IPDPS*, 2008.
[21] O. Younis and S. Fahmy, “Distributed clustering in ad-hoc sensor networks: a hybrid, energy-efficient approach”, *IEEE Trans. Mobile Computing*, Vol. 3, No. 4, 2004.
[22] Panasonic Ni-MH battery handbook, “http://www2.renovaar.ee/userfiles/Panasonic_Ni-MH_Handbook.pdf”.
[23] Y. Wang, M. C. Vuran and S. Goddard, “Stochastic analysis of energy consumption wireless sensor networks,” *Proc. of IEEE Secon*, 2010.
[24] R. Nelson, *Probability, Stochastic Processes, and Queueing Theory: The Mathematics of Computer Performance Modeling*, Springer, 1995.
[25] S. Ross, *A First Course in Probability*, 8th Ed, Prentice Hall, 2009.
[26] C. Miller, A. Tucker and R. Zemlin,“Integer programming formulations and traveling salesman problems,” *Journal of the ACM*, pp. 326-329, 1960.
[27] B. Gavish, “A note on the formulation of the m-salesman traveling salesman problem,” Management Science, 1976.
[28] Tolga Bektas, “The multiple traveling salesman problem: an overview of formulations and solution procedures,”*Intl Journal of Management Science*, pp. 209-219, 2006.
[29] M. M. Solomon, “Algorithm for the vehicle routing and scheduling problems with time window constraints,”*Operation Research*, vol. 35, No. 2, pp.254 - 265, 1987.
[30] M.W.P. Savelsbergh, “Local search in routing problems with time windows,” *Annals of Operation Research*, pp. 285-305, 1985.
[31] Snezana Mitrovic-Minic, Ramesh Krishnamurti, “The multiple TSP with time windows: vehicle bounds based on precedence graphs,” *Operations Research Letters*, pp. 111 - 120, 2006.
[32] M. Desrochers, J.K. Lenstra, M.W.P. Savelsbergh and F. Soumis, “Vehicle routing with time windows: optimization and approximation,” Elsevier Science Publisher, 1988.
[33] H.C. Lau, M. Sim, K. M. Teo,“Vehicle routing problem with time windows and a limited number of vehicles”,*European Journal of Operation Research*, 148, pp. 559 - 569, 2003.
[34] E. Taillard, P. Badeau, M. Gendreau, F. Geurtin, J.Y. Potvin, “A tabu search heuristic for the vehicle routing problem with soft time windows,”*Transportation Science*, vol. 31, pp. 170- 186, 1997.
[35] N. Bansal, A. Blum, S. Chawla and A. Meyerson, “Approximation algorithms for Deadline-TSP and Vehicle Routing with Time Windows,” *ACM STOC*, 2004.
[36] M. Gendreau, A. Hertz, G. Laporte and M. Stan, “A generalized insertion heuristic for the traveling salesman problem with time windows,” *Journal Operations Research*, vol. 46 no. 3, 1998, pp. 330-335.
[37] A. Tanenbaum, *Modern Operating Systems*, pp. 125, 3rd Ed, Prentice Hall, 2007.
[38] W. R. Heinzelman, A. Chandrakasan and H. Balakrishnan, “Energy-efficient communication protocol for wireless microsensor networks,” *IEEE HICSS*, 2000.

Cong Wang received the BEng degree in Information Engineering from the Chinese University of Hong Kong and M.S. degree in Electrical Engineering from Columbia University, New York. He is currently working towards the PhD degree at the Department of Electrical and Computer Engineering, Stony Brook University, New York. His research interests include wireless sensor networks, performance evaluation of network protocols and algorithms.

Ji Li received the B.S. degree in Electrical Engineering from Harbin Engineering University, Harbin, China, and the M.S. degree in Electrical Engineering from Zhejiang University, Hangzhou, China. He is currently a PhD student in the Department of Electrical and Computer Engineering, Stony Brook University, Stony Brook, New York. His research interests include wireless sensor networks and embedded systems.



Fan Ye received his B.E. and M.S. degrees in Automation and Computer Science from Tsinghua University, Beijing, China, and Ph.D. in Computer Science from UCLA. He then joined IBM T. J. Watson Research as a Research Staff Member, working on stream processing systems, cloud messaging and mobile computing. He served as the Chair for IBM Watson Mobile Computing Professional Interest Community for two years. He is currently an associate professor with the Center for Energy-efficient Computing and Applications at the EECS school of Peking University, Beijing, China. He holds over a dozen US/international patents and patent applications. His research interests include mobile computing, mobile cloud, wireless networks, sensor networks and their applications.

Yuanyuan Yang received the BEng and MS degrees in computer science and engineering from Tsinghua University, Beijing, China, and the MSE and PhD degrees in computer science from Johns Hopkins University, Baltimore, Maryland. She is a professor of computer engineering and computer science at Stony Brook University, New York, and the director of Communications and Devices Division at New York State Center of Excellence in Wireless and Information Technology (CEWIT). Her research interests include wireless networks, data center networks, optical networks and high-speed networks. She has published more than 270 papers in major journals and refereed conference proceedings and holds seven US patents in these areas. She is currently the Associate Editor-in-Chief for the IEEE Transactions on Computers and an Associate Editor for the Journal of Parallel and Distributed Computing. She has served as an Associate Editor for the IEEE Transactions on Computers and IEEE Transactions on Parallel and Distributed Systems. She has served as a general chair, program chair, or vice chair for several major conferences and a program committee member for numerous conferences. She is an IEEE Fellow.