Cross-Layer Sleep Scheduling Design in Service-Oriented Wireless Sensor Networks

Jianping Wang, *Member*, *IEEE*, Deying Li, *Member*, *IEEE*, Guoliang Xing, *Member*, *IEEE*, and Hongwei Du, *Member*, *IEEE*

Abstract—Service-oriented wireless sensor networks have recently been proposed to provide an integrated platform, where new applications can be rapidly developed through flexible service composition. In wireless sensor networks, sensors are periodically switched into the sleep mode for energy saving. This, however, will cause the unavailability of nodes, which, in turn, incurs disruptions to the service compositions requested by the applications. Thus, it is desirable to maintain enough active sensors in the system to provide each required service at any time in order to achieve dependable service compositions for various applications. In this paper, we study the cross-layer sleep scheduling design, which aims to prolong the network lifetime while satisfying the service availability requirement at the application layer. We formally define the problem, prove that the problem is NP-hard, and develop two approximation algorithms based on the LP relaxation and one efficient reordering heuristic algorithm. The proposed work will enhance the dependability of the service composition in service-oriented wireless sensor networks.

Index Terms—Service-oriented architecture, wireless sensor network, sleep scheduling, service composition.

1 Introduction

Wireless Sensor Networks (WSNs) hold the promise of revolutionizing the way we observe and interact with the physical world in a wide range of application domains. Most recently, it has been realized that it is hard to rapidly develop new applications in the WSNs designed for domain-specific applications. The main reason for this is the lack of both standard operations and representation for sensor data that can be used by diverse sensor applications [1], [2]. To support the interoperability between different applications, service-oriented architectures (SOAs) for wireless sensor networks have been proposed [2], [3], [4], [5], [6], [8].

Service-oriented WSN architecture treats software components provided by sensors as services [6], e.g., time synchronization services, node localization services, data aggregation services, and data processing services [7]. As a result, new applications can be rapidly developed through flexible service composition. In a service-oriented WSN, each service can be provided by multiple nodes. A typical application consists of a set of required services that need to work coordinately through service composition. The task of service composition is to assign each required service to an appropriate node according to certain criteria. Service composition with various performance metrics [9], [10], [11], [12], e.g., load balance, end-to-end delay, guaranteed bandwidth and resource, have been well studied within the context of the Internet. Service composition in wireless sensor networks has also recently been studied in [13], [14]. In particular, Abrams and Liu [13] study the minimum-cost service placement based on service composition graphs with a tree structure. Srivastava et al. [14] consider the optimal placement of filters (services) with different selectivity rates so that the cost of execution and communication is minimized.

Prolonging network life is a primary consideration in the design of WSNs. A major power conservation technique in wireless sensor networks is to design adaptive sleep schedules at the MAC layer to minimize the energy consumption due to idle listening [15], [16]. However, a sensor in the sleep mode is not able to communicate, which, in turn, affects the operations at both the network layer and the application layer. Thus, a cross-layer sleep scheduling design is desired to devise sleep scheduling, which considers the requirements at other layers such as sensing coverage, delay efficiency, energy consumption, and network connectivity [17], [18], [19], [20], [21].

In a service-oriented WSN, the task of service compositions brings unique challenges to sensor sleep scheduling. Consider a surveillance application that uses video sensors to capture raw images [14]. Suppose that the user is looking for images in which the monitored area is dimly lit and there is a lot of motion between successive frames (indicating suspicious activity). Three services are required in this application: service s_1 checking for dim images by calculating the average pixel intensity, service s_2 checking for "sufficient" motion between successive frames, and service s_3 fusing the identified motions to arrive at a high-level result (e.g., the appearance of a suspect). Suppose that there are four sensors (as shown in Fig. 1) which can provide services $\{s_1, s_3\}$, $\{s_2\}$, $\{s_3\}$, and $\{s_1, s_2\}$, respectively. If all those four sensors are active simultaneously, the

- *J. Wang is with the Department of Computer Science, City University of Hong Kong, Kowloon, Hong Kong. E-mail: jianwang@cityu.edu.hk.*
- *D. Li is with the Key Laboratory of Data Engineering and Knowledge Engineering, Renmin University of China, Beijing 100872, China. E-mail: deyingli@ruc.edu.cn.*
- *G. Xing is with the Department of Computer Science and Engineering, Michigan State University, MI 48824. E-mail: glxing@msu.edu.*
- *H. Du is with the Department of Computer Science, Illinois Institute of Technology, Chicago, IL 60616. E-mail: hongwei.du@ieee.org.*

Manuscript received 18 Mar. 2008; revised 3 Sept. 2009; accepted 17 Mar. 2010; published online 28 June 2010.
For information on obtaining reprints of this article, please send e-mail to: tmc@computer.org, and reference IEEECS Log Number TMC-2008-03-0092.
Digital Object Identifier no. 10.1109/TMC.2010.124.

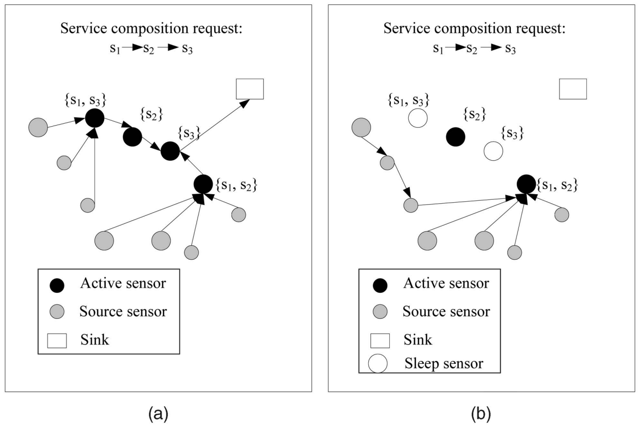


Fig. 1. The impact of sleep scheduling on service composition.

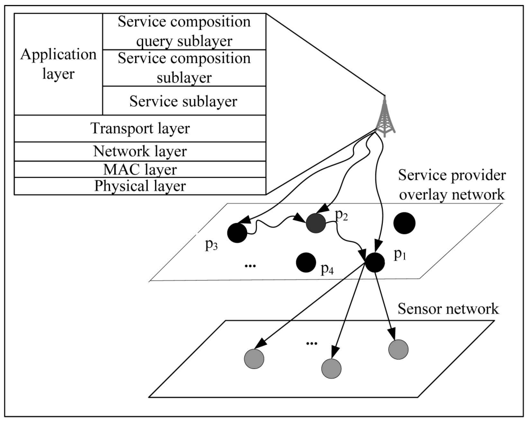


Fig. 2. System model.

network can accommodate the service composition request as shown in Fig. 1a. If only one of the sensors is in the sleep mode, then it can be verified that the service composition can still be feasibly accommodated. However, if the two sensors providing s_3 are scheduled to be in the sleep mode at the same time, as shown in Fig. 1b, then the service composition request cannot be accommodated until one of the sensors providing service s_3 wakes up, which causes the disruption to the service composition.

Fig. 1 suggests that in a service-oriented WSN, sensor sleep scheduling should be able to satisfy the service availability requirement at the application layer, i.e., to guarantee that there are enough active sensors providing each potential required service at any time. This paper will focus on designing such *service-availability-aware* sleep scheduling schemes. To the best of our knowledge, this is the first work to design adaptive sleep scheduling scheme considering both the service availability requirements at the application layer and the need of prolonging the network lifetime.

We will consider two sources of energy consumption for a sensor that can be reduced through sleep scheduling. The first source, also the dominate one, is the energy consumption wasted on idle listening. To minimize such energy consumption, we will try to switch a sensor into the sleep mode as much as possible. The second source is the energy consumption caused by the wake-up operations when a sensor is switched from the sleep mode to the active mode. As shown in [21], a sequence of operations must be performed when a sensor switches from the sleep mode to the active mode ready for receiving/transmitting. Such operations include initializing the radio by a timer interrupt and waiting for the radio's crystal oscillator to stabilize. The energy consumed by all these operations is referred to as wake-up energy consumption. Clearly, in order to minimize the wake-up energy consumption, we prefer less frequent mode changes for a sensor. Our sleep scheduling algorithms address both sources of the energy consumption.

We summarize our contributions as follows:

- We propose the *service-availability-aware* sleep scheduling for service-oriented WSNs, formally define the problems, and prove that the problems are NP-hard.
- We propose an LP-relaxation-based approximation algorithm with the worst-case bound analysis for the case without considering the wake-up energy consumption.
- We propose an LP-relaxation-based approximation algorithm with the worst-case bound analysis for the case considering the wake-up consumption. We further develop another heuristic algorithm to reduce the wake-up energy consumption for a given sleep schedule.
- We analyze the impact of cycle length on sleep scheduling.

The rest of the paper is organized as follows: The problem definition and complexity analysis are given in Section 2. In Section 3, we approach the problem for a special case by assuming that the wake-up energy consumption is negligible. We then solve the more general problem by considering the wake-up energy consumption in Section 4. We further discuss the impact of the cycle length in Section 5. The computational study is conducted in Section 6. We conclude the paper in Section 7.

2 Problem Description

In this section, we first introduce the system model, then present the problem definition, and analyze the problem complexity.

2.1 System Model

In our network architecture, the service providers form a service provider overlay network as shown in Fig. 2. The application layer at the sink of the service-oriented WSN has the following three layers:

- *Service composition query sublayer*. Service composition query sublayer maps an application's query into a *service composition query*, which specifies the required services and their invocation order. For example, the aforementioned query reporting the image with potential suspects will be converted into a service composition query which requests that image information is processed with services s_1, s_2, and s_3 before it is sent to the sink.
- *Service sublayer*. Service sublayer has the service information provided by the sensors in the service provider overlay network.

- *Service composition sublayer*. Service composition layer finds the service composition solutions for service composition queries based on the information from *service composition query sublayer* and *service sublayer*.

The proposed work in this paper is to derive the sleep schedule for the sensors in the service provider overlay network given the information at the service sublayer of the sink. The sleep schedule designed in this paper aims to prolong the network longevity while maintaining enough active sensors for each required service from the application layer. The proposed sleep schedule is not designed to satisfy the service requirements of a particular application. Instead, it is designed to satisfy the service requirements of the system, where multiple applications developed in such a service-oriented WSN may run simultaneously. Given such a long-term systemwise objective, we believe that a centralized sleep scheduling design that aims to prolong the network longevity while satisfying the system's requirements on active service providers for each service is worth of study.

The sleep scheduling studied in this paper can be integrated with other sleep scheduling schemes available in the literature [17], [18], [19], [20], [21] to satisfy other system performance requirements such as sensing coverage, network connectivity, and delay efficiency. In this paper, we only consider the sleep scheduling design for the sensors in the service provider overlay network and we assume that there is a path between any two service providers in the service provider overlay network. Without loss of generality, sensors in the rest of the paper refer to the sensors that can provide services in service-oriented WSNs.

Suppose that in a service-oriented WSN, there are m services $S = \{s_1, s_2, \ldots, s_m\}$ provided by n sensors $P = \{p_1, p_2, \ldots, p_n\}$. We assume that each sensor p_i can provide a subset of the services. We use S_i to denote the set of services that sensor p_i can provide, and P_j be the set of sensors that can provide service s_j. Each sensor will be alternatively in active and sleep modes. If a sensor p_i is active, it can provide all services in S_i; if it is sleeping, it cannot provide any services.

We rotate the active/sleep modes of the sensors based on a cyclic schedule, where each cycle contains T equal-length time slots. In each time slot t, a sensor p_i is either active or asleep. In this paper, we assume that all sensors in the service provider overlay network are globally synchronized. Fig. 3 demonstrates how the proposed sleep scheduling scheme works. In the example, there are four sensors providing $\{s_1, s_2, s_3\}$, $\{s_1, s_2\}$, $\{s_2, s_3\}$, and $\{s_1, s_3\}$, respectively. Then we can derive a sleep schedule as shown in Fig. 3 with cycle length $T = 4$. Under the sleep schedule given in Fig. 3, p_1 only needs to be active in one time slot of a cycle, p_2 needs to be active in two consecutive time slots of a cycle, p_3 needs to be active in two separate time slots of a cycle, and p_4 needs to be active in two consecutive time slots of a cycle. Such a schedule guarantees the service availability at all times.

In a service-oriented WSN, multiple service composition requests may exist concurrently. As a result, some services may be required by multiple service composition requests, and consequently, need more active sensors, which can provide such services. In particular, we assume that each service s_j needs d_j active sensors at any time slot in order to process the service composition requests in the system, which can be derived according to the historical information of the service composition requests at the application layer. For example, according to the historical information, if there are at most five applications simultaneously requesting service s_1 at any time in the system, d_1 is 5. For the example shown in Fig. 3, suppose that each service requires one active sensor at any time, then the sleep schedule in Fig. 3 is a feasible sleep schedule.

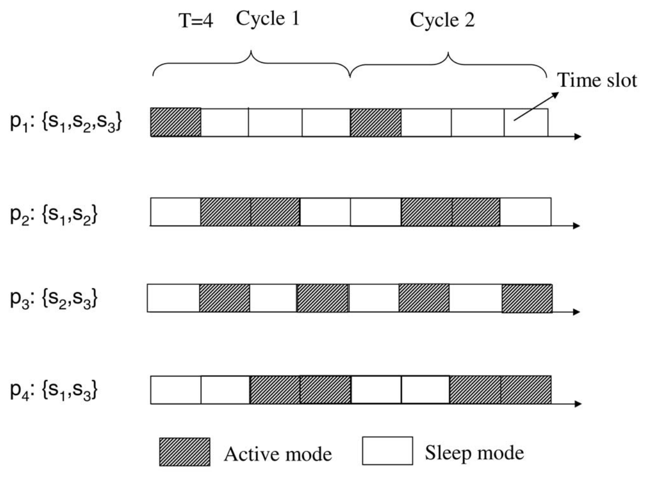


Fig. 3. An illustrative example of the proposed sleep scheduling scheme.

2.2 Problem Definition

The problem studied in this paper is to determine a sleep schedule for each sensor in P so that there are enough number of active sensors for each service s_j at any time. Let x_{it} be a binary variable, where $x_{it} = 1$ indicates that sensor p_i is active at time slot t. Then we need to guarantee that $\sum_{p_i \in P_j} x_{it} \geq d_j$ for any service s_j at any time slot t.

Under a sleep schedule $\{x_{it}\}$, each sensor p_i has a load

$$\alpha_i = \sum_{t=1}^{T} x_{it}, \tag{1}$$

i.e., the number of active time slots within a cycle, which indicates the energy consumption for p_i. Essentially, we hope to reduce α_i to save energy since the energy consumption is a key concern in wireless sensor networks. Our specific objective is to minimize the maximum load, $\max\{\alpha_i\}$, for the purpose of load balancing. Such an objective will help to extend the longevity of the network as a whole. We use **P1** to refer to such a problem.

Based on problem **P1**, we will further study a more general problem **P2**, which considers the extra energy consumption when a sensor switches from the sleep mode to the active mode. Given a sleep schedule $\{x_{it}\}$, we can use a binary variable y_{it} to indicate such a mode switch at the beginning of time slot t, where $y_{it} = \max\{x_{it} - x_{it-1}, 0\}$ for $t = 2, \ldots, T$ and $y_{i1} = \max\{x_{i1} - x_{iT}, 0\}$. Note that $y_{it} = \max\{x_{it} - x_{it-1}, 0\}$ indicates that $y_{it} = 1$ if and only if $x_{it-1} = 0$ and $x_{it} = 1$ for $1 < t \leq T$. Since we assume that the sleep scheduling is cyclic, if $x_{iT} = 0$ and $x_{i1} = 1$, then the first time slot in a new cycle will incur the wake-up energy consumption; otherwise, there is no wake-up energy consumption at the beginning of the first time slot. Thus, $y_{i1} = \max\{x_{i1} - x_{iT}, 0\}$.

If we normalize the energy consumption for an active sensor in one time slot to be 1, then we can assume that the energy consumption for a sensor to switch from the sleep mode to the active mode is B. For example, according to [21], the wake-up energy consumption is roughly $15mA * 3V * 0.7ms = 31.5\mu J$ and the minimum energy consumption of idle listening in an active period is $15mA * 3V * 115ms = 5175\mu J$ in S-MAC. If we normalize the energy consumption in an active period to be 1, then the wake-up energy consumption B is 0.006. Note that if the length of a time slot is short, i.e., all services can be finished in a short time, for example, in a low data rate WSN [22], B could be large. Thus, it is important to consider the wake-up energy consumption and merge active time slots as much as possible to reduce the energy consumption while satisfying the requirements of enough active sensors in each time slot.

Considering the wake-up energy consumption, the total energy consumption for a sensor p_i in a cycle is

$$\beta_i = \sum_{t=1}^{T} x_{it} + B \sum_{t=1}^{T} y_{it}. \tag{2}$$

As a result, the objective of problem **P2** is to minimize the maximum β_i, $\max\{\beta_i\}$. Although, in general, we have $B < 1$, it is important to include the impact of B, even if $B \ll 1$ (e.g., [21]), because the sleep schedule will be repeatedly executed for a long time.

In both problems **P1** and **P2**, we assume that the cycle length T is externally given. As a matter of fact, the cycle length T also plays an important role in an energy-efficient sleep schedule. Therefore, after solving problems **P1** and **P2** under a given T, we will further discuss how to determine the value of T. The details will be elaborated later.

2.3 Problem Complexity

Before presenting our algorithms for problems **P1** and **P2**, we first show that the problems are computationally intractable. Because problem **P1** is a special case of problem **P2**, we only need to show that problem **P1** is NP-hard.

Theorem 1. *Problem **P1** is NP-hard in the strong sense even with $T = 2$.*

Proof. We prove the theorem by a reduction from the Not-All-Equal 3SAT (NAE3SAT) problem, which is known as NP-complete in the strong sense.

NAE3SAT: For a given set $U = \{u_1, u_2, \dots, u_n\}$ of n binary variables, and a collection of clauses $C_1, \dots, C_m$ defined over the literals $\{u_1, u_2, \dots, u_n, \bar{u}_1, \bar{u}_2, \dots, \bar{u}_n\}$ such that each clause has three literals $C_i = u_{i1} \vee u_{i2} \vee u_{i3}$, the question is whether there exists a true assignment for $C = C_1 \wedge C_2 \wedge \cdots \wedge C_m$ such that each clause C_i has at least one true literal and at least one false literal.

Given an instance of NAE3SAT, we can construct an instance of the decision version of problem **P1** as follows: Let there be $2n$ sensors each corresponding to a literal. With a bit of abuse of the notation, we also denote the sensors as $\{u_1, u_2, \dots, u_n, \bar{u}_1, \bar{u}_2, \dots, \bar{u}_n\}$. Let there be $n + m$ services, where

1. each service s_i, $i = 1, \dots, n$, can be provided by two sensors $\{u_i, \bar{u}_i\}$ and
2. each service s_{n+i}, $i = 1, \dots, m$, can be provided by three sensors $\{u_{i1}, u_{i2}, u_{i3}\}$, where $C_i = u_{i1} \vee u_{i2} \vee u_{i3}$. Let each service need one sensor at any time.

The question is whether there is a sleep scheduling with $T = 2$ and $\max\{\alpha_i\} \leq 1$.

1. Suppose that there is a true assignment for NAE3SAT. Then we can have a sleep schedule with $T = 2$. In the assignment, if $u_i = 1$ (implying $\bar{u}_i = 0$), let sensor u_i be active at $t = 1$ and sleeping at $t = 2$, and sensor $\bar{u}_i$ be sleeping at $t = 1$ and active at $t = 2$; if $u_i = 0$ (implying $\bar{u}_i = 1$), let sensor u_i be sleeping at $t = 1$ and active at $t = 2$, and sensor $\bar{u}_i$ be active at $t = 1$ and sleeping at $t = 2$.

   In such a schedule, for any service s_i, $i = 1, \dots, n$, one and only one of the sensors $\{u_i, \bar{u}_i\}$ will be active at any time t; for any service s_{n+i}, $i = 1, \dots, m$, because one of the literals $\{u_{i1}, u_{i2}, u_{i3}\}$ is true, at least one sensor in $\{u_{i1}, u_{i2}, u_{i3}\}$ is active at $t = 1$; because one of the literals $\{u_{i1}, u_{i2}, u_{i3}\}$ is false, at least one sensor in $\{u_{i1}, u_{i2}, u_{i3}\}$ is active at $t = 2$. Therefore, the sleep schedule is feasible. Because each sensor is alternatively active and sleeping, we have $\alpha_i = 1$ for all sensors.
2. Suppose that there is a sleep schedule with $T = 2$ and $\max\{\alpha_i\} \leq 1$. So any sensor cannot be active for more than one time slot in a cycle. Since service s_i, for $i = 1, \dots, n$, needs one sensor from $\{u_i, \bar{u}_i\}$ at any time, both sensors u_i and $\bar{u}_i$ must be alternatively active for one time slot. Meanwhile, at $t = 1$, for any service s_{n+i}, $i = 1, \dots, m$, the three feasible sensors u_{i1}, u_{i2}, and u_{i3} cannot be all active; for otherwise, there will be no active sensors for service v_{n+i} at $t = 2$.

In the corresponding NAE3SAT problem, we can let $u_i = 1$ if sensor u_i is active at time $t = 1$, and $u_i = 0$ if sensor u_i is active at time $t = 2$. The above analysis shows that this is a feasible solution to NAE3SAT. □

3 Sleep Scheduling without Considering the Wake-up Energy Consumption

In this section, we approach problem **P1**, which minimizes the maximum energy consumption among all sensors without considering the wake-up energy consumption in a given cycle with length T. This section is organized as follows: We first formulate the problem into an integer linear programming (ILP) problem, then present a round-up heuristic algorithm which is based on linear programming relaxation, followed by an improved round-up heuristic algorithm.

3.1 ILP Formulation

For the convenience of discussion, we reintroduce our notations as follows:

i: index of sensors, where $1 \leq i \leq n$.
j: index of services, where $1 \leq j \leq m$.

```
Step 1: Solve the LP relaxation Eq.(4)-(6) and (8)
  to obtain a fractional solution $\{x_{it}^L\}$.
Step 2: For each $t \in \{1, 2, \ldots, T\}$
          For each $j \in \{1, \ldots, m\}$
            (a) set $x_{it}^H = 1$ for the $d_j$ largest $x_{it}^L$
                in $\{x_{it}^L | p_i \in P_j\}$, and
            (b) set other $x_{it}^H = 0$
```

Fig. 4. Simple round-up algorithm for problem **P1**.

t: index of time slots, where $1 \leq t \leq T$.
We define binary variable

$$x_{it} = \begin{cases} 1, & \text{if sensor } p_i \text{ is active at time } t; \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

Then we have an ILP formulation as follows:

$$\min \alpha \tag{4}$$

subject to:

$$\sum_{p_i \in P_j} x_{it} \geq d_j, \quad \forall 1 \leq j \leq m, \quad 1 \leq t \leq T, \tag{5}$$

$$\sum_{t=1}^{T} x_{it} \leq \alpha, \quad \forall 1 \leq i \leq n, \tag{6}$$

$$x_{it} = \{1, 0\}, \quad \forall 1 \leq i \leq n, \ 1 \leq t \leq T \tag{7}$$

In the formulation, constraint (5) ensures that each service s_j will be provided by at least d_j active sensors at any time, and (6) evaluates the maximum energy consumption among all sensors. Recall that we have defined the energy consumption of a sensor p_i as $\alpha_i = \sum_{t=1}^{T} x_{it}$ in (1). So (6) is actually equivalent to $\sum_{t=1}^{T} x_{it} = \alpha_i \leq \alpha$ for $1 \leq i \leq n$ (where α is to be minimized), though there is no need to explicitly include α_i in the formulation.

3.2 Simple Round-Up Algorithm Based on Linear Programming Relaxation

Because of the NP-hardness of problem **P1**, we design an approximation algorithm based on the linear programming (LP) relaxation of the above ILP. The algorithm contains two stages. In the first stage, we solve its LP relaxation by removing the integral constraints (7), i.e., (7) is replaced by

$$0 \leq x_{it} \leq 1, \forall 1 \leq i \leq n, 1 \leq t \leq T. \tag{8}$$

In the second stage, a greedy round-up algorithm is employed to find an integral solution based on the solution obtained in the first stage. We use x_{it}^L to represent the optimal solution to the LP relaxation, where some x_{it}^L may not be integers, and use x_{it}^H to represent the solution obtained based on rounding x_{it}^L. In the round-up algorithm presented in Fig. 4, we consider the sleep schedule time slot by time slot. In each time slot t, for each service s_j, the first d_j sensors which can provide s_j and have the highest x_{it}^L will be selected to be active at time slot t.

It is known that an LP can be solved efficiently. The sorting of the elements in $\{x_{it}^L\}$ takes $O(n \log n)$ time for each t and j. So the time complexity in Step 2 is $O(Tmn \log n)$, which is dominated by solving the LP relaxation.

Lemma 1. *For any $x_{it}^H = 1$, we must have $x_{it}^L \geq \frac{1}{r}$, where $r = \max_{1 \leq j \leq m}\{|P_j| - d_j + 1\}$.*

Proof. We prove it by contradiction. If not, i.e., there is an $x_{it}^H = 1$, but $x_{it}^L < \frac{1}{r}$. Since $x_{it}^H = 1$ is among d_j largest x_{it}^L for $p_i \in P_j$, there are at most $d_j - 1$ sensors such that $x_{it}^L \geq \frac{1}{r}$. Consequently, there are at least $|P_j| - d_j + 1$ sensors such that $x_{it}^L < \frac{1}{r}$. So,

$$\sum_{i \in P_j} x_{it}^L < d_j - 1 + \frac{|P_j| - d_j + 1}{r} \leq d_j. \tag{9}$$

It contradicts that $\{x_{it}^L\}$ is feasible to the LP relaxation. □

Theorem 2. *The round-up algorithm for problem **P1** has an approximation ratio r, where $r = \max_{1 \leq j \leq m}\{|P_j| - d_j + 1\}$.*

Proof. Suppose that α^H is the objective function value obtained by the round-up algorithm, α^L is the optimal objective function value for the LP relaxation, and α^* is the optimal objective function value.

By Lemma 1, when $x_{it}^H = 1$, we have $x_{it}^L \geq \frac{1}{r} = \frac{1}{r} x_{it}^H$, and when $x_{it}^H = 0$, we have $x_{it}^L \geq 0 \geq \frac{1}{r} x_{it}^H$. Therefore, we have $x_{it}^H \leq r x_{it}^L$, for all i and t. Then,

$$\begin{aligned} \alpha^H &= \max_{1 \leq i \leq n} \left\{ \sum_{t=1}^{T} x_{it}^H \right\} \\ &\leq \max_{1 \leq i \leq n} \left\{ \sum_{t=1}^{T} r x_{it}^L \right\} \\ &= r \max_{1 \leq i \leq n} \left\{ \sum_{t=1}^{T} x_{it}^L \right\} \\ &= r\alpha^L \leq r\alpha^*. \end{aligned} \tag{10}$$

□

3.3 Improved Round-Up Algorithm Based on Linear Programming Relaxation

The round-up algorithm can be further improved as follows: Noting that in Step 2 of rounding, for each service s_j and time t, we independently select d_j sensors and set $x_{it}^H = 1$. However, in doing this, we may have set too many sensors to be active because we have ignored the fact that each active sensor can provide multiple services. For example, suppose that a service s_j has selected to round up x_{it}^L to $x_{it}^H = 1$ for a sensor p_i at time slot t. When we consider another service $s_{j'}$ with $p_i \in P_{j'}$, x_{it}^L may not be selected to be rounded up by service $s_{j'}$. As a result, there may be more active sensors than $d_{j'}$ for service $s_{j'}$.

To have a better rounding scheme, and at the same time, to maintain the approximation ratio derived in Theorem 2, we improve the rounding scheme by solving a second LP relaxation. In this second LP relaxation, if a sensor p_i is not active at time slot t under the round-up algorithm, we will force it to be in the sleep mode in the new LP; if a sensor p_i is active under the round-up algorithm, we will reconsider it. Specifically, we fix $x_{it} = 0$ if $x_{it}^H = 0$ in the round-up algorithm, and then solve LP (4) to (6) with the following new constraint (11):

$$0 \leq x_{it} \leq x_{it}^H, \forall 1 \leq i \leq n, 1 \leq t \leq T. \tag{11}$$

```
Step 1: Solve the LP relaxation Eq.(4)-(6) and (8)
        to obtain the optimal solution $\{x_{it}^L\}$.
Step 2: (a) for each $t$, each $j$, set $x_{it}^H = 1$ for the $d_j$
            largest $x_{it}^L$ in $\{x_{it}^L | p_i \in P_j\}$, and
        (b) set other $x_{it}^H = 0$.
Step 3: Solve the LP relaxation Eq.(4)-(6) and (11)
        to obtain the optimal solution $\{x_{it}^{L2}\}$.
Step 4: (a) for each $t$, each $j$
            set $x_{it}^{H2} = 1$ for the $d_j$ largest $x_{it}^{L2}$
            in $\{x_{it}^{L2} | p_i \in P_j\}$, and
        (b) set other $x_{it}^{H2} = 0$.
```

Fig. 5. Improved round-up algorithm for problem **P1**.

Let x_{it}^{L2} be the optimal solution for the above LP. Then we will repeat the same rounding scheme. The complete algorithm is described in Fig. 5.

Clearly, we have $x_{it}^{H2} \le x_{it}^H$ because $x_{it}^{H2} = 1$ only when $x_{it}^H = 1$. As a result, the new objective function value under x_{it}^{H2} will be smaller than or equal to α^H.

4 Sleep Scheduling with the Consideration of Wake-Up Energy Consumption

In this section, we consider the case of including the wake-up energy consumption when a sensor changes its status from sleep to active. Without loss of generality, we have assumed that the energy consumption of being active for one time slot is one, and the wake-up energy consumption is B. Then the cycle-wide energy consumption for each sensor is the sum of the number of active time slots and the total wake-up energy consumption. Our objective is to minimize the maximum cycle-wide energy consumption among all sensors. We also assume that the energy consumption for each active time slot is larger than one wake-up energy consumption, i.e., $0 \le B < 1$.

The problem can be formulated into an ILP with new binary variables y_{it}

$$y_{it} = \begin{cases} 1, & \text{if sensor } p_i \text{ has a wake-up at the} \\ & \text{beginning of the time slot } t, \\ 0, & \text{otherwise.} \end{cases}$$

Then, by definition, we have

$$y_{it} = \max\{x_{it} - x_{it-1}, 0\}, \quad \forall 1 \le i \le n, 2 \le t \le T.$$

To evaluate the possible wake-up at $t = 1$, we need

$$y_{i1} = \max\{x_{i1} - x_{iT}, 0\}, \quad \forall 1 \le i \le n.$$

The above two constraints can be combined into

$$y_{it} = \max\{x_{it} - x_{it-1}, 0\}, \quad \forall 1 \le i \le n, 1 \le t \le T$$

under the understanding that x_{i0} represents $x_{i,T}$. Then we have the following ILP to minimize the maximum cycle-wide energy consumption for all sensors:

$$\min \beta, \tag{12}$$

$$\sum_{p_i \in P_j} x_{it} \ge d_j, \quad \forall 1 \le j \le m, 1 \le t \le T, \tag{13}$$

$$\sum_{t=1}^{T} x_{it} + B \sum_{t=1}^{T} y_{it} \le \beta, \quad \forall 1 \le i \le n, \tag{14}$$

$$x_{it} - x_{i,t-1} \le y_{it}, \quad \forall 1 \le i \le n, 1 \le t \le T, \tag{15}$$

$$x_{it} \in \{0,1\}, y_{it} \in \{0,1\} \quad \forall \quad 1 \le i \le n, 1 \le t \le T. \tag{16}$$

```
Step 1: Solve the LP relaxation (12)-(15) and (17) to
        obtain an optimal solution $\{x_{it}^L, y_{it}^L\}$.
Step 2: (a) for each $t$, each $j$, set $x_{it}^H = 1$ for $d_j$ largest
            $x_{it}^L$ in $\{x_{it}^L | p_i \in P_j\}$;
        (b) set other $x_{it}^H = 0$, and
        (c) compute $y_{it}^H$ by using $x_{it}^H$.
```

Fig. 6. Round-up algorithm for problem **P2**.

Recall that we have defined the total energy consumption of a sensor p_i as $\beta_i = \sum_{t=1}^{T} x_{it} + B \sum_{t=1}^{T} y_{it}$ in (2). So (14) is actually equivalent to $\sum_{t=1}^{T} x_{it} + B \sum_{t=1}^{T} y_{it} = \beta_i \le \beta$, though we do not explicitly include β_i in the formulation.

4.1 Round-Up Heuristic Algorithm

In the following, we will propose an approximation algorithm similar to the round-up algorithm for problem **P1**, i.e., based on an LP relaxation. It has two stages, where in the first stage, we find an optimal solution for the LP relaxation, and in the second stage, a greedy algorithm is employed to find an integral solution based on the solution obtained in the first stage.

In the LP relaxation, we replace (16) by

$$0 \le x_{it}, y_{it} \le 1 \quad \forall \quad 1 \le i \le n, 1 \le t \le T. \tag{17}$$

Again, we use $\{x_{it}^L, y_{it}^L\}$ to represent the optimal solution to the LP relaxation. Now, we consider the second stage of rounding the fractional values of $\{x_{it}^L, y_{it}^L\}$ to integral solutions $\{x_{it}^H, y_{it}^H\}$. We follow the same rounding procedure as before to round $\{x_{it}^L\}$ to $\{x_{it}^H\}$, and then calculate $\{y_{it}^H\}$ based on $\{x_{it}^H\}$. The algorithm is formally presented in Fig. 6.

Similarly as Theorem 2, the round-up algorithm for problem **P2** has an approximation ratio $2r$, where $r = \max_{1 \le j \le m}\{|P_j| - d_j + 1\}$. We omit the proof here.

While the round-up algorithm for problem **P2** has a theoretical worst-case bound, it may not be able to find reasonably good solutions for some cases. We have observed that in the LP relaxation, the y_{it} term in (15) will make $x_{it} = x_{i,t-1}$ whenever possible. As a result, for some sensor i, x_{it} may have the same value for all $1 \le t \le T$. In the round-up stage, such sensor may be selected to be active in all time slots, which results in a very poor solution.

We thus have developed another different heuristic algorithm based on the assumption that $B \le 1$. The algorithm works well when B is much smaller than 1, a reasonable assumption in many cases. In the new heuristic algorithm, we first assume $B = 0$ and use the improved round-up algorithm in Fig. 5 to obtain a solution without considering wake-up energy consumption. Then we reorder the schedule to reduce the wake-up cost. The details are presented in the next section.

4.2 Heuristic Algorithm for Reordering a Sleep Schedule

We use a vector $X_t^H = (x_{1t}^H, x_{2t}^H, \ldots, x_{nt}^H)^T$ to denote the status of all sensors at time slot t. Suppose that we have obtained a sleep schedule $X^H = (X_1^H, X_2^H, \ldots, X_T^H)$ by the improved round-up algorithm for the case without considering the wake-up energy consumption. Note that X^H is a matrix where row i represents sensor p_i's sleep schedule over time t and column j represents the status of all sensors in time slot t_j in X^H.

We notice that any reordering of X_i^H and X_j^H in X^H, for example, $(X_2^H, X_1^H, X_3^H, \ldots, X_T^H)$, can also feasibly provide enough active sensors to satisfy the system requirement on the number of active sensors at any time slot. However, the reordering will affect the wake-up energy consumption of some sensors. We thus aim to develop a heuristic approach to reduce the wake-up energy consumption by reordering the columns in X^H.

Because our objective is to minimize the maximum cycle-wide energy consumption, including both being active and being activated from the sleep mode among all sensors, one greedy idea to achieve such a goal is to let those sensors with higher energy consumption for being active, i.e., with heavier load, have fewer wake-ups. This can be done by reordering the columns in X^H.

Let $\alpha_i = \sum_{t=1}^T x_{it}^H$ be the load of sensor p_i in the given schedule. Then we sort the sensors in the nonincreasing order of their loads. We use $[j]$ to denote the jth sensor in the sorting, i.e., $\alpha_{[1]} \geq \alpha_{[2]} \geq \cdots \geq \alpha_{[n]}$. Then we are to reorder the columns in X^H under which the wake-up energy consumption of $p_{[j]}$ is no more than that of sensor $p_{[j+1]}$, for $j = 1, \ldots, n-1$. In other words, we need a sorting of $(X_1^H, X_2^H, \ldots, X_T^H)$ in which sensor $p_{[j]}$ should have a higher priority than sensor $p_{[j+1]}$ to reduce the wake-up energy consumption.

We first consider sensor $p_{[1]}$. To ensure the minimum wake-up energy consumption for $p_{[1]}$, $(X_1^H, X_2^H, \ldots, X_T^H)$ should be split into two subsets Y_1^+ and Y_1^-, where

$$Y_1^+ = \{X_t^H | x_{[1],t}^H = 1, t = 1, 2, \ldots, T\},$$
$$Y_1^- = \{X_t^H | x_{[1],t}^H = 0, t = 1, 2, \ldots, T\},$$

and all X_t^H in Y_1^+ are consecutively scheduled before all X_t^H in Y_1^-. In such a way, sensor $p_{[1]}$ wakes up at most once, regardless of the sequences within Y_1^+ and Y_1^-, though the sequences within Y_1^+ and Y_1^- will affect the wake-up energy consumption for other sensors.

Now consider sensor $p_{[2]}$ after the determination of Y_1^+ and Y_1^-. To ensure the minimum wake-up energy consumption for $p_{[2]}$ under Y_1^+ and Y_1^-, we can further split Y_1^+ into Y_2^{++} and Y_2^{+-}, and split Y_1^- into Y_2^{--} and Y_2^{-+} such that

$$Y_2^{++} = \{X_t^H | x_{[2],t}^H = 1, X_t^H \in Y_1^+\},$$
$$Y_2^{+-} = \{X_t^H | x_{[2],t}^H = 0, X_t^H \in Y_1^+\},$$
$$Y_2^{-+} = \{X_t^H | x_{[2],t}^H = 1, X_t^H \in Y_1^-\},$$
$$Y_2^{--} = \{X_t^H | x_{[2],t}^H = 0, X_t^H \in Y_1^-\}.$$

Then all X_t^H will be scheduled in the group order of Y_2^{++}, Y_2^{+-}, Y_2^{--}, and Y_2^{++} such that sensor $p_{[2]}$ wakes up at most once. At the same time, such a schedule does not change the wake-up energy consumption for sensor $p_{[1]}$ because

$$Y_2^{++} \cup Y_2^{+-} = Y_1^+, Y_2^{--} \cup Y_2^{-+} = Y_1^-.$$

In general, when we consider sensor $p_{[j]}$, all X_t^H can be classified into up to 2^j subsets in the form of Y_j^σ, where σ is a string of $\{+,-\}$ with j bits. For a $\sigma = \sigma' \cup \{+\}$ or $\sigma = \sigma' \cup \{-\}$ with σ' being $j-1$ bits, Y_j^σ is recursively defined as

$$Y_j^{\sigma' \cup \{+\}} = \{X_t^H | x_{[j],t}^H = 1, X_t^H \in Y_{j-1}^{\sigma'}\},$$
$$Y_j^{\sigma' \cup \{-\}} = \{X_t^H | x_{[j],t}^H = 0, X_t^H \in Y_{j-1}^{\sigma'}\}.$$

We denote $|\sigma| = 1$ if σ contains even number of "$-$", and $|\sigma| = 0$ if σ contains odd number of "$-$".

Given Y_j^σ which is defined based on $Y_{j-1}^{\sigma'}$, we can schedule all X_t^H in the same subset Y_j^σ in an arbitrary sequence. The groupwise sequence of Y_j^σ is recursively determined by $Y_{j-1}^{\sigma'}$, where 1) $Y_j^{\sigma' \cup \{+\}}$ and $Y_j^{\sigma' \cup \{-\}}$ are consecutive and 2) their orders are determined by $|\sigma'|$ such that $Y_j^{\sigma' \cup \{+\}}$ precedes $Y_j^{\sigma' \cup \{-\}}$ if $|\sigma'| = 1$, and $Y_j^{\sigma' \cup \{+\}}$ follows $Y_j^{\sigma' \cup \{-\}}$ if $|\sigma'| = 0$.

Because the number of subsets may increase quickly with j, we can terminate the process after a predefined j_0. In doing this, we guarantee that the j_0 sensors with the heaviest load have fewer wake-ups than the remaining $n - j_0$ sensors. This is reasonable in a heuristic algorithm because we can tolerate a light-load sensor to have more wake-ups. The pseudocode of the algorithm is given in Fig. 7.

```
for $i = 1$ to $n$ do
  $\alpha_i = \sum_{t=1}^T x_{it}^H$
sort the sensors into the non-increasing order of $\alpha_i$, i.e.
$\alpha_{[1]} \geq \alpha_{[2]} \ldots \geq \alpha_{[n]}$
$Y_1^+ = \{X_t^H | x_{[1],t}^H = 1, t = 1, 2, \ldots, T\}$
$Y_1^- = \{X_t^H | x_{[1],t}^H = 0, t = 1, 2, \ldots, T\}$
$\Sigma_1 = \{``+", ``-"\}$
Initially let the columns in $Y_1^+$ precede the columns
in $Y_1^-$, i.e., $Y_1^+ \rightarrow Y_1^-$
for $j = 2$ to $j_0$ do
  for each $\sigma \in \Sigma_{j-1}$ do
    $Y_j^{\sigma \cup \{+\}} = \{X_t^H | x_{[j],t}^H = 1, X_t^H \in Y_{j-1}^\sigma\}$
    $Y_j^{\sigma \cup \{-\}} = \{X_t^H | x_{[j],t}^H = 0, X_t^H \in Y_{j-1}^\sigma\}$
    $\Sigma_j = \Sigma_j \cup \{\sigma \& \{+\}\}$
    $\Sigma_j = \Sigma_j \cup \{\sigma \& \{-\}\}$
    if $|\sigma| = 1$ then
      $Y_j^{\sigma \cup \{+\}} \rightarrow Y_j^{\sigma \cup \{-\}}$, otherwise
      $Y_j^{\sigma \cup \{-\}} \rightarrow Y_j^{\sigma \cup \{+\}}$
The orders of the columns in $Y_{j_0}^\sigma$ where $\sigma \in \Sigma_{j_0}$ can be
arbitrary. The union of $Y_{j_0}^\sigma$ according to the defined
sequence forms a new sleep scheduling solution.
```

Fig. 7. Reordering algorithm for problem **P2**.

We now use an example to illustrate the above approach. Suppose that we have a solution X^H with $n = 5$ and $T = 6$ as shown in Table 1. If we set $j_0 = 3$, then the three sensors with the heaviest load are $[1] = 1$, $[2] = 2$, and $[3] = 3$. Y_j^σ can

TABLE 1
One Example X^H for the Reordering Algorithm

X_1^H	X_2^H	X_3^H	X_4^H	X_5^H	X_6^H
1	0	1	0	1	1
1	0	0	1	0	1
0	1	0	0	1	0
0	0	1	0	0	0
0	1	0	0	0	0

TABLE 2
Result of Reordering X^H

X_1^H	X_6^H	X_3^H	X_5^H	X_2^H	X_4^H
1	1	1	1	0	0
1	1	0	0	0	1
0	0	0	1	1	0
0	0	1	0	0	0
0	0	0	0	1	0

be calculated as given in Fig. 8. Consequently, we will reorder the sleep schedule as

$$\left(X_1^H, X_6^H, X_3^H, X_5^H, X_2^H, X_4^H\right).$$

In such a schedule, all $p_{[1]}$, $p_{[2]}$, $p_{[3]}$ only have one wake-up in one cycle as shown in Table 2.

5 Determining the Cycle Length T and the Time Slot Length

5.1 On the Cycle Length

So far, we have assumed that the cycle length T is given and our objective is to minimize the energy consumption within the given cycle. To find a more energy-efficient sleep scheduling, we can further include the cycle length T as a decision variable. To compare the energy consumption under different cycle lengths, we will consider the long-term average energy consumption of a sensor. Specifically, suppose that in a particular sleep schedule π with cycle length T, sensor p_i has an energy consumption $\alpha_i(\pi;T)$ within a cycle T, then we define

- $\eta_i(\pi;T) = \alpha_i(\pi;T)/T$ as the long-term average energy consumption of sensor p_i when the sleep schedule π is repetitively implemented.
- $\eta^*(\pi;T) = \max_i\{\eta_i(\pi;T) \mid i = 1,\dots,n\}$ as the maximum average energy consumption among all sensors under π.
- $\eta^*(T) = \min_\pi\{\eta^*(\pi;T)\}$ as the minimized maximum average consumption among all sleep schedules with cycle length T.

Then our objective becomes to minimize $\eta^*(T)$, that is, to find an η^* such that

$$\eta^* = \min_T\{\eta^*(T) \mid T = 2,3,\dots,T_{\max}\},$$

where $T_{\max}$ is the maximum cycle length under consideration. Note that $\eta^*(T)$ with a given T has been studied in the previous sections. We now study the problem of finding η^* by optimizing over T. The impact of T can be shown by the following examples.

Suppose that there are four sensors dedicated to providing service s_1 that needs three sensors at any time. If we set $T = 2$, then there are at least two sensors that have to be active in both time slots, which leads to $\eta^*(2) = 1$. If we set $T = 4$, then there is a solution where there is exactly one different sensor being sleep in each time slot. Thus, $\eta^*(4) = 3/4$. This example shows that a longer cycle length may lead to a lower average energy consumption because there will be more freedom for optimization. In fact, for any cycle length T and positive integer k, we have

$$\eta^*(T) \geq \eta^*(kT)$$

because given a feasible sleep schedule π for a cycle with length T, we can always repeat πk times to get a feasible schedule, denoted by π^k, for the cycle with length kT, where the two schedules have the same average load for each sensor, or $\eta^*(\pi;T) = \eta^*(\pi^k;kT)$.

However, $\eta^*(T)$ is not necessarily a decreasing function of T. This can be shown by the following example: Suppose that there are two sensors p_1 and p_2 dedicated to providing service s_1 that needs one sensor at any time. If we set $T = 2$,

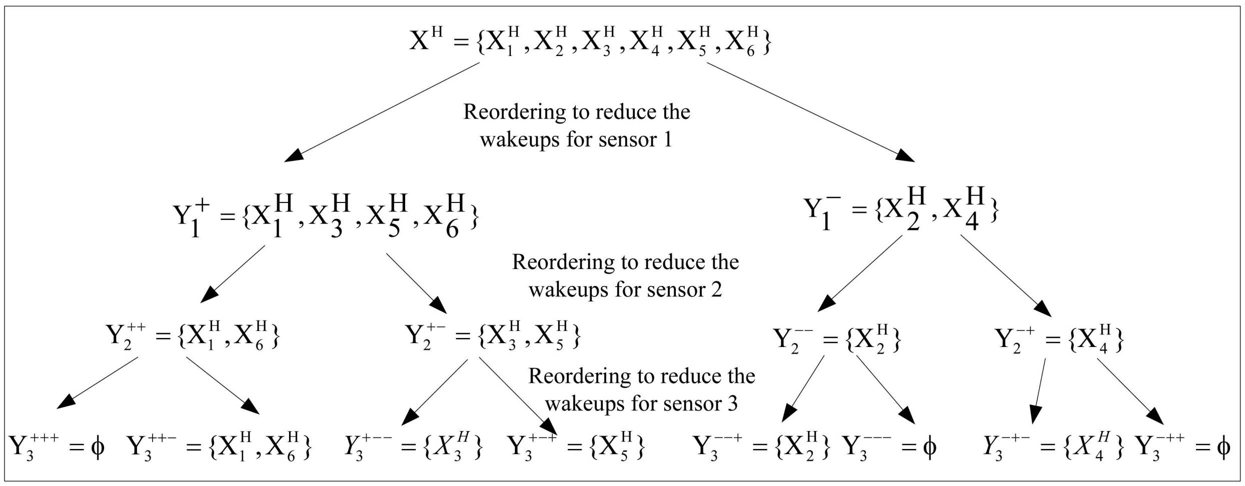

$X^H = \{X_1^H, X_2^H, X_3^H, X_4^H, X_5^H, X_6^H\}$
Reordering to reduce the wakeups for sensor 1
$Y_1^+ = \{X_1^H, X_3^H, X_5^H, X_6^H\}$
$Y_1^- = \{X_2^H, X_4^H\}$
Reordering to reduce the wakeups for sensor 2
$Y_2^{++} = \{X_1^H, X_6^H\}$
$Y_2^{+-} = \{X_3^H, X_5^H\}$
$Y_2^{--} = \{X_2^H\}$
$Y_2^{-+} = \{X_4^H\}$
Reordering to reduce the wakeups for sensor 3
$Y_3^{+++} = \phi$
$Y_3^{++-} = \{X_1^H, X_6^H\}$
$Y_3^{+--} = \{X_3^H\}$
$Y_3^{+-+} = \{X_5^H\}$
$Y_3^{--+} = \{X_2^H\}$
$Y_3^{---} = \phi$
$Y_3^{-+-} = \{X_4^H\}$
$Y_3^{-++} = \phi$

Fig. 8. Reordering procedure for the example in Table 1.

then the two sensors p_1 and p_2 can be active alternatively, which leads to $\eta^*(2) = 0.5$. However, if we set $T = 3$, then we have to set one sensor to be active for two time slots, which leads to $\eta^*(3) = 2/3$, i.e., $\eta^*(T)$ may become larger with the increase of T.

Despite the above fact, we have observed in our computational experiments that $\eta^*(T)$ follows a general decreasing trend of T. Actually, we have

$$\eta^*(T+1) \leq \frac{T}{T+1}\eta^*(T) + \frac{1}{T+1} \tag{18}$$

because a feasible schedule to a cycle of length T will lead to a feasible schedule to a cycle with length $T+1$ by being appended with a time slot in which all sensors are active. Due to (18), when $T_{\max}$ is large enough, we can derive the sleep scheduling design with the cycle length of $T_{\max}$ instead of exhaustively searching for the optimal T.

5.2 On the Time Slot Length

Finally, we make a brief discussion about the impact of the time slot length. In our modeling framework, we assume that there is a given time slot length, where the energy consumption of an active time slot is normalized into one and the wake-up energy consumption is B (relative to the energy consumption of one active time slot).

For such a model, one may argue that if we increase the time slot length to k times and redefine the energy consumption of the new long single time slot to be one, then the wake-up energy becomes B/k. This will have no impact on average power consumption rate and the service availabilities. As k increases, the impact of the wake-up energy consumption diminishes, and hence, problem **P2** degenerates to problem **P1**. In other words, we always only need to solve the simpler problem **P1** and make the time slot longer enough to make the wake-up energy consumption relatively negligible.

However, we also have incentive to use short time slot due to the fact that a cycle with more time slots can generally have a better schedule than a cycle with less number of time slots. Specifically, for any schedule with T time slots, if we are able to reschedule it based on half of the original time slot, subject to some hardware or software availability constraints, then we have a problem of $2T$ time slots, which potentially has a better schedule, though the wake-up energy consumption doubles relative to the new time slot length.

In summary, the above discussion shows that there may exist certain trade-off between the time slot length and the cycle length (i.e., the number of time slots), correlated by the relative value of the wake-up energy consumption. Without digging into too much theoretical analysis on this, we will provide some computational study to show their connection in the next section.

6 Simulation

In this section, we demonstrate the effectiveness of our heuristic algorithms through simulation. We first introduce the design of our simulation. For a wireless sensor network with n sensors providing m services, we can generate a problem instance as follows: For each service s_j, we randomly determine its requirement on the number of active sensors at any time slot from a discrete uniform distribution $d_j \in \{5, 6, \ldots, 10\}$. For each pair of sensor p_i and service s_j, we randomly determine whether p_i is able to provide service s_j with a probability $q, 0 < q < 1$. We then test whether it is feasible to support the requirements of all d_j. If infeasible, we drop the instance from our experiments. We have tested different values of q, and obtained similar observations. Thus, here, we only report the results of $q = 0.5$.

TABLE 3
Relative Error of the Improved Round-Up Algorithm for Problem **P1** Subject to the Optimal Schedule

(n,m)	8	16	32	64	128	256	512
$(30,10)$	0.039	0.037	0.031	0.028	0.027	0.022	0.019
$(30,20)$	0.040	0.041	0.043	0.034	0.028	0.027	0.024
$(30,30)$	0.045	0.055	0.061	0.042	0.038	0.036	0.032
$(30,40)$	0.076	0.069	0.062	0.045	0.039	0.038	0.037

For each given combination of the parameter values of n and m, we randomly generate 500 feasible independent problem instances as above, and solve them under different values of T. We have tested T values of the powers of 2, i.e., $T \in \{8, 16, 32, \ldots, 512\}$. All our results are the average over the 500 instances with given n, m, and T.

6.1 Performance of Problem P1

In the first set of computational experiments, we test the effectiveness of the improved round-up algorithm for problem **P1**. When the problem sizes are small, for example, with $n = 30$, we find that ILP (4)-(7) can be optimally solved by CPLEX MIP solver within 10 minutes for most cases. Therefore, we can compare the results of our heuristic solutions with the optimal ones. In Table 3, we report the relative error of the improved round-up algorithm for problem **P1** subject to the optimal ILP solution.

From Table 3, we can see that the relative errors of our heuristic solutions become smaller with the increase of the cycle length T. There are at least two possible explanations for such a trend. First, as in many ILP problems, the LP relaxation will generally become close to ILP when the problem sizes become large. So, the round-up heuristic solution based on an LP relaxation will also become close to the optimal solution when T is large. Second, when T is small, the optimal solution is also small. Thus, a small difference will lead to a large relative error. For example, when $T = 8$, the optimal solution α^* may be 5 or 6. In such a case, one difference in the heuristic solution will lead to a relative error of 0.2 or 0.167.

When the problems sizes become large, for example, with $n = 40$ or $n = 50$, the ILP (4)-(7) cannot be efficiently solved by CPLEX MIP solver except when T is small. For such large problems, we compare our improved round-up algorithm with a lower bound, which is simply the optimal objective function value of the LP relaxation. The relative errors subject to the lower bound are given in Table 4.

From Table 4, we have obtained similar observations that the relative errors become smaller with the increase of T. Summarizing the results in both tables, we also find that the relative errors increase with the number of sensors n and the number of services m. With the increase of n, there will be higher possibility of making a mistake when a roundup

TABLE 4
Relative Error of the Improved Round-Up Algorithm for Problem **P1** Subject to the Lower Bound

(n, m)	8	16	32	64	128	256	512
(40,20)	0.105	0.127	0.087	0.072	0.053	0.044	0.040
(40,30)	0.162	0.128	0.086	0.072	0.057	0.054	0.044
(40,40)	0.211	0.151	0.088	0.074	0.064	0.057	0.048
(40,50)	0.136	0.133	0.107	0.088	0.074	0.066	0.054
(50,20)	0.145	0.126	0.097	0.078	0.059	0.051	0.041
(50,30)	0.179	0.165	0.134	0.102	0.081	0.069	0.054
(50,40)	0.223	0.200	0.145	0.110	0.099	0.076	0.062
(50,50)	0.262	0.206	0.179	0.138	0.107	0.087	0.074

is conducted for each service. Thus, the relative error increases with the increase of n. With the increase of m, there are more roundups to be conducted, which will also cause a larger error. Nevertheless, the real performance of our heuristic solutions may be better because the relative errors in Table 4 are subject to lower bounds rather than the optimal solutions.

6.2 Performance of Problem P2

In the second set of our experiments, we study the effectiveness of our heuristic solution for problem **P2**. Recall that our heuristic solution is obtained by the improved round-up algorithm for problem **P1** followed by a reordering procedure. We first show how the reordering procedure may help improve the solution quality. For this end, we will compare the solution difference between **P1** and **P2** (with and without reordering), under different wake-up energy consumption values B, where we use the term "relative error" to refer to the relative extra cost of a solution without reordering over the cost of the solution with reordering. The results are given in Figs. 9 and 10.

In Fig. 9, we report the relative errors without the reordering procedure for the case of $n = 30$ and $m = 30$. We see that the relative error without reordering generally increases with the wake-up energy B, which is understandable. Moreover, the error is large when the cycle length T is large, which can be roughly explained by the fact that a longer cycle length contains more switches between sleep and wake-up, and hence, it is of more importance to do reordering.

In Fig. 10, we report the relative errors without the reordering procedure for the case of $n = 50$ and $m = 30$. While Fig. 10 shows the similar trend of larger errors with

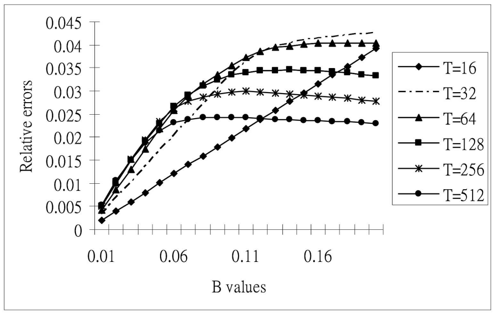


Fig. 10. Relative error of solutions without reordering for problem **P2** with 50 sensors and 30 services.

the increase of B values, it does reveal some different behaviors regarding the cycle length T. When B is large, the cases with shorter cycles $T = 16$ and $T = 32$ tend to have larger errors than the cases with longer cycles $T = 256$ and $T = 512$. We note that there are more sensors in Fig. 10 than in Fig. 9. Consequently, each sensor will need less frequent switches with sleep and wake-up in Fig. 10. This may partially explain the difference in the two figures.

From both figures, we see that the solution without reordering has a relative error no more than 4 percent for most cases. This may suggest that the round-up algorithm can be used as an acceptable approximate approach when B is small. On the other hand, however, the reordering approach is indeed able to improve the solution quality by a small margin. Since the reordering can be done quickly, it should be taken as a helpful postprocessing step.

We now report the effectiveness of the solution quality after the reordering step. We find that the ILP formulation (12)-(16) for problem **P2** is much harder to solve, where CPLEX MIP solver cannot find optimal solutions within 10 minutes even with $n = 30$ when T is large. Therefore, we will compare our heuristic solution with the lower bound obtained from LP relaxation. The results are given in Figs. 11 and 12.

In Fig. 11, we study the case of $n = 30$ and $m = 30$ with different values of T. We vary $B \in \{0, 0.01, 0.02, \ldots, 0.20\}$. We see that the relative error increases with the increase of B's value. When B is smaller, e.g., less than 0.10, the relative errors increase slowly with B. This shows the effectiveness of our heuristic approach for those practical cases with small B values.

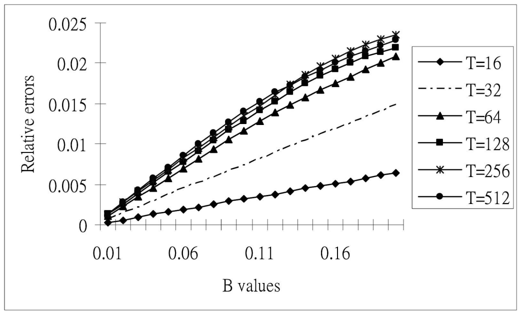


Fig. 9. Relative error of solutions without reordering for problem **P2** with 30 sensors and 30 services.

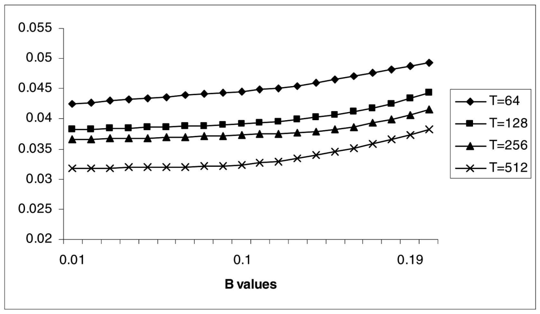


Fig. 11. Relative error of the reordering algorithm for problem **P2** with 30 sensors and 30 services.

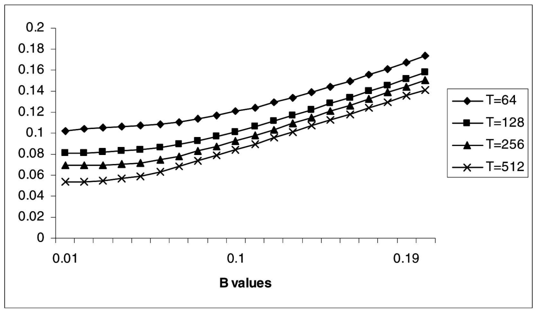


Fig. 12. Relative error of the reordering algorithm for problem **P2** with 50 sensors and 30 services.

To further validate the insights obtained from Fig. 11, we have also tested the case of $n = 50$ and $m = 30$. The results are given in Fig. 12. For such large size problems, the relative errors increase more quickly with B, but we can still observe a slow increase with small B values. This reconfirms the effectiveness of our heuristic approach for those practical cases with small B values.

6.3 On the Cycle Length and Time Slot Length

We now study in details the impact of the cycle length T on the long-term average energy consumption. We have selected three scenarios $(n = 40, m = 20)$, $(n = 50, m = 20)$, and $(n = 60, m = 10)$. For each scenario, a single problem instance is generated and optimally solved under different values of $T \in \{4, 5, \ldots, 40\}$. The average energy consumption $\eta^*(T)$ for each scenario is plotted in Fig. 13. Fig. 13 illustrates a general downside trend of $\eta^*(T)$ with T. It also clearly shows that $\eta^*(T)$ may periodically become large with the increase of T. This is consistent with our discussion on the impact of T. We see that the curves of $\eta^*(T)$ tend to converge with the increase of T. Such an observation partially supports our suggestion that we can derive the sleep scheduling design with the cycle length of $T_{\max}$, if $T_{\max}$ is large enough, instead of exhaustedly searching the optimal T.

We finally show some impact of the time slot length in Fig. 14. We use the case of $T = 16$ as the benchmark. For each wake-up energy consumption value B, we consider to use half, quarter, and one-eighth of the time slot length for rescheduling, which leads to the cases of $T = 32$ with wake-up energy $2B$, $T = 64$ with wake-up energy $4B$, and $T = 128$ with wake-up energy $8B$, respectively. We then show the solution value relative to the benchmark with $T = 16$.

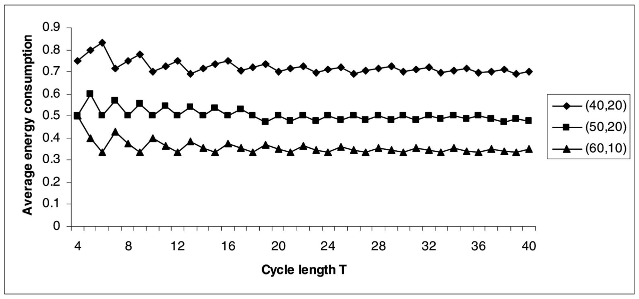


Fig. 13. The impact of cycle length.

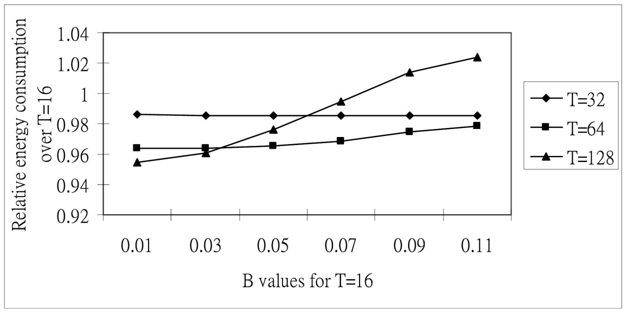


Fig. 14. Impact of the time slot length.

Fig. 14 clearly shows the benefit of using shorter time slots, especially with smaller B values. However, the benefit is not unbounded. For example, we see that the case of $T = 128$ will have a higher cost than the benchmark when B is large, which implies that we should not make the time slot shorter if the wake-up energy consumption is already very high relative to the single time slot consumption.

7 Conclusion

This paper studies the cross-layer sleep scheduling design in service-oriented WSNs. We aim to design a sleep scheduling scheme, which can minimize the energy consumption and guarantee that there are enough active sensors providing each required service in the system at any time. We prove that such a *service-availability-aware* sleep scheduling problem is NP-hard. We then consider a special case of the problem without considering the wake-up energy consumption. We propose one approximation algorithm based on LP relaxation for such a special case. For the general case of the problem, we propose one approximation algorithm based on LP relaxation and one heuristic algorithm which reorders the sleep schedule for the special case. We also consider the impact of the cycle length on the sleep scheduling design. Our simulation results demonstrate the effectiveness of the proposed algorithms.

In our sleep scheduling design, we have assumed that each service has a known requirement on the number of active sensors based on the historical service composition requests in the system, and our goal is to design a long-term plan for sleep scheduling to better support the service compositions at the application layer. With the sleep scheduling designed in this work, when a service composition request arrives at the system, a feasible/optimal service composition solution can be derived by assigning each required service in the request to an active sensor. Such a *sleep-scheduling-aware* service composition is under investigation in parallel to this paper.

Acknowledgments

The work described in this paper was supported in part by grants from the Research Grants Council of the Hong Kong Special Administrative Region, China (Project No. CityU 121107), the National Science Foundation of China under grant no. 10671208, the 863 High-Tech Project under grant no. 2008AA01Z120, and Renmin University of China under grant no. 10XNJ032.

References

[1] J. Liu and F. Zhao, "Towards Semantic Services for Sensor-Rich Information Systems," *Proc. Second IEEE/CreateNet Int'l Workshop Broadband Advanced Sensor Networks (Basenets '05),* Oct. 2005.

[2] X. Chu and R. Buyya, "Service Oriented Sensor Web," *Sensor Network and Configuration: Fundamentals, Techniques, Platforms, and Applications Experiments,* N.P. Mahalik, ed., Springer-Verlag, 2006.

[3] D. Gračanin, M. Eltoweissy, A. Wadaa, and L. DaSilva, "A Service-Centric Model for Wireless Sensor Networks," *IEEE J. Selected Areas Comm.,* vol. 23, no. 6, pp. 1159-1166, June 2005.

[4] J. Liu, E. Cheong, and F. Zhao, "Semantics-Based Optimization across Uncoordinated Tasks in Networked Embedded Systems," *Proc. ACM Int'l Conf. Embedded Software (EMSOFT '05),* pp. 272-281, Sept. 2005.

[5] X. Koutsoukos, M. Kushwaha, I. Amundson, S. Neema, and J. Sztipanovits, "Service-Oriented Architectures for Networked Embedded Sensor Systems," *Proc. Conf. Monterey,* Oct. 2006.

[6] J. King, R. Bose, S. Pickles, A. Helal, S. Ploeg, and J. Russo, "Atlas: A Service-Oriented Sensor Platform," *Proc. First IEEE Int'l Workshop Practical Issues in Building Sensor Network Applications (SenseApp '06),* Nov. 2006.

[7] S. Sundresh, G. Agha, K. Mechitov, W. Kim, and Y. Kwon, "Coordination Services for Wireless Sensor Networks," *Proc. Int'l Workshop Advanced Sensors, Structural Health Monitoring and Smart Structures,* 2003.

[8] K. Aberer, M. Hauswirth, and A. Salehi, "Global Sensor Networks," Technical Report LSIR-REPORT-2006-001, 2006.

[9] B. Raman and R.H. Katz, "Load Balancing and Stability Issues in Algorithms for Service Composition," *Proc. IEEE INFOCOM,* pp. 1477-1487, 2003.

[10] J. Jin and K. Nahrstedt, "Source-Based QoS Service Routing in Distributed Service Networks," *Proc. IEEE Int'l Conf. Comm. 2004 (ICC '04),* pp. 20-24, June 2004.

[11] X. Gu, K. Nahrstedt, R. Chang, and C. Ward, "QoS-Assured Service Composition in Managed Service Overlay Networks," *Proc. 23rd IEEE Int'l Conf. Distributed Computing Systems (ICDCS '03),* pp. 19-22, May 2003.

[12] J. Jin and K. Nahrstedt, "Resource- and Quality-Aware Application-Level Service Multicast," *Proc. Ninth IEEE Int'l Workshop Future Trends of Distributed Computing Systems (FTDCS '03),* pp. 198-204, May 2003.

[13] Z. Abrams and J. Liu, "Greedy Is Good: On Service Tree Placement for In-Network Stream Processing," *Proc. Int'l Conf. Distributed Computing Systems (ICDCS),* 2006.

[14] U. Srivastava, K. Munagala, and J. Widom, "Operator Placement for In-Network Stream Query Processing," *Proc. 24th ACM Symp. Principles of Database Systems (PODS '05),* June 2005.

[15] W. Ye, J. Heidemann, and D. Estrin, "An Energy-Efficient MAC Protocol for Wireless Sensor Networks," *Proc. IEEE Conf. Computer Comm.,* vol. 3, pp. 1567-1576, June 2002.

[16] T. Dam and K. Langendoen, "An Adaptive Energy-Efficient MAC Protocol for Wireless Sensor Networks," *Proc. ACM Conf. Networked Sensor Systems,* pp. 171-180, Nov. 2003.

[17] G. Lu, N. Sadagopan, B. Krishnamachari, and A. Goel, "Delay Efficient Sleep Scheduling in Wireless Sensor Networks," *Proc. IEEE INFOCOM,* vol. 4, pp. 2470-2481, 2005.

[18] R. Ha, P.-H. Ho, and X.S. Shen, "SS-Trees: A Cross-Layer Organizational Approach for Mesh-Based Wide-Area Wireless Sensor Networks," *Proc. IEEE Int'l Conf. Broadband Networks (BroadNets),* pp. 885-894, Oct. 2005.

[19] R. Ha, P.-H. Ho, and X.S. Shen, "Cross-Layer Organization for Wireless Sensor Networks Using Sense-Sleep Trees," *Proc. Int'l Conf. Wireless Networks, Comm. and Mobile Computing (WirelessCom '05),* pp. 952-957, June 2005.

[20] R. Ha, P.-H. Ho, and X.S. Shen, "Cross-Layer Application-Specific Wireless Sensor Network Design with Single-Channel CSMA MAC over Sense-Sleep Trees," *Computer Comm.,* vol. 29, pp. 3425-3444, 2006.

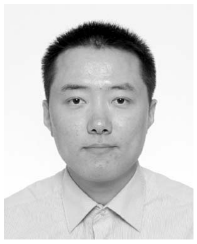
[21] J. Polastre, J. Hill, and D. Culler, "Versatile Low Power Media Access for Wireless Sensor Networks," *Proc. Int'l Conf. Embedded Networked Sensor Systems (SenSys '04),* 2004.

[22] J. Ma, W. Lou, Y. Wu, X. Li, and G. Chen, "Energy Efficient TDMA Sleep Scheduling in Wireless Sensor Networks," *Proc. IEEE INFOCOM,* 2009.

Jianping Wang received the BSc and MSc degrees in computer science from Nankai University, Tianjin, China, in 1996 and 1999, respectively, and the PhD degree in computer science from the University of Texas at Dallas in 2003. She is currently an assistant professor in the Department of Computer Science, City University of Hong Kong. Her research interests include wireless networks, optical networks, and network coding. She is a member of the IEEE.

Deying Li received the MS degree in mathematics from Huazhong Normal University in 1988 and the PhD degree in computer science from the City University of Hong Kong in 2004. She is currently a professor in the Department of Computer Science, Renmin University of China. Her research interests include wireless networks, mobile computing, and algorithm design and analysis. She has published more than 80 research papers in various prestigious journals such as the *IEEE Transactions on Mobile Computing*, the *IEEE Transactions on Parallel and Distributed Systems*, the *IEEE Transactions on Computers*, *Theoretical Computer Science*, the *Journal of Parallel and Distributed Computing*, *Computer Communications*, *Computer Journal*, *Networks*, *Journal of Combinatorial Optimization*, *Discrete Mathematics*, *Journal of Graph Theory*, and conferences such as IEEE INFOCOM, ICDCS, ICC, and MASS. She is a member of the IEEE.

Guoliang Xing received the BS degree in electrical engineering and the MS degree in computer science from Xi'an Jiao Tong University, China, in 1998 and 2001, respectively, and the MS and DSc degrees in computer science and engineering from Washington University in St. Louis, in 2003 and 2006, respectively. He is an assistant professor in the Department of Computer Science and Engineering at Michigan State University. From 2006 to 2008, he was an assistant professor of computer science at the City University of Hong Kong. He served on a number of technical program committees and held several workshop organization positions including the program cochair of the First ACM International Workshop on Heterogeneous Sensor and Actor Networks (HeterSanet 2008) and the Workshop on Wireless Ad Hoc and Sensor Networks (WWASN 2008 and 2009). His research interests include the design and analysis of fusion-based sensor networks, interference in low-power wireless networks, and performance control of cyber-physical systems. He is a recipient of the US National Science Foundation CAREER Award in 2010. He is a member of the IEEE.



Hongwei Du received the BSc degree from the Central China Normal University in 2003 and the PhD degree from the City University of Hong Kong in 2008. He is currently a senior research associate in the Department of Computer Science at Illinois Institute of Technology and also a postdoctor in the Academy of Mathematics and System Sciences, Chinese Academy of Sciences. His research interests include wireless networking, mobile computing, and algorithm design. He is a member of the IEEE and the ACM.