Efficient Power Management for Wireless Sensor Networks: a Data-Driven Approach

MingJian Tang[1], Jinli Cao[2], and Xiaohua Jia[3]
[1,2]Department of Computer Science and Computer Engineering, La Trobe University, Melbourne, Australia
[3]Department of Computer Science, City University of Hong Kong, Kowloon, Hong Kong
Email: [1]mj2tang@students.latrobe.edu.au, [2]j.cao@latrobe.edu.au, and [3]jia@cs.cityu.edu.hk

***Abstract*—Providing energy-efficient continuous data collection services is of paramount importance to Wireless Sensor Network (WSN) applications. This paper proposes a new power management framework called Data-Driven Power Management (DDPM) as the infrastructure for integrating various energy efficient techniques, such as the approximate querying and the sleep scheduling. By utilizing the beneficial properties of these techniques, we can achieve better energy efficiency while still meeting the application specific criteria, such as data accuracy and communication latency. The distinguishing feature of DDPM is that it starts by exploiting the natural tradeoff between the quality of the sensor data and the energy consumption, and then it generates a precision-guaranteed estimation for each sensor node as its maximum sleep time. Eventually deterministic schedules can be made by the DDPM based on these estimations. We further propose two decentralized algorithms so that the undesirable communication delays caused by staggered local sleep schedules can be avoided. The experimental results show that the nodes' sleep times can be significantly increased while incurring only a minor rise in latency.**

Keywords-approximate query; multi-objective optimization; sleep scheduling; wireless sensor networks

I. INTRODUCTION

The unprecedented focus on Wireless Sensor Networks (WSNs) has made feasible for a wide range of applications [1], such as environmental monitoring, military surveillance, and intelligent information gathering. Typically a WSN consists of a large number of battery-powered sensor nodes and a base station. These nodes are densely distributed into the physical area and continuously or periodically feed the base station with the collected data [13]. After deployment, the replenishments of node batteries become either too expensive or impossible. Additionally, the relatively slow advancements in the battery technology make the energy constraint the fundamental challenge in the WSN. As a consequence, power management has become strongly imperative.

Reducing the in-network communications is one of the primary aspects of WSN power management since radio communication dominates the power consumption in the WSN. Approximate querying [4, 12, 14] is one of the most popular techniques that exploits the natural tradeoffs between energy consumption and data accuracy. The technique basically relies on the application-specific error bound which are disseminated to each sensor node along with the query. The rationale behind the technique is that a query result is only sent back to the base station if the change of two consecutive sensor values violates a user-defined error bound. Since it is widely accepted and observed that tight correlations largely exist in WSN applications, using these error bounds essentially exploits this nodal-level correlation (temporal correlation). By taking advantage of approximate querying, the frequency of the results transmissions can be reduced, while the data accuracy can still be guaranteed. In [4], a data-collection protocol, which is precision-constrained based, is proposed for sensor environments. The paper essentially revolves around quality-aware data collection protocols that enable quality requirements of the queries to be satisfied while minimizing the energy consumption. In [12], the author further exploits the spatial suppression in addition to the temporal suppression. The rationale behind the paper is the constraint chaining, which builds a network of constraints that are maintained locally and yet allows a global view of values to be maintained with minimal cost. In [14], the author discusses the optimal error bound assignment to each node with respect to maximizing the WSN lifetime. Although the approximate query is a powerful power management paradigm, the existing approaches do not further explore the energy-saving opportunities for the suppressed nodes.

Reducing the energy consumption in idle state is another important aspect of WSN power management since energy consumed by the radio circuit in idle state is almost equal to that consumed in active state. In order to prolong the network lifetime, it is necessary to actively schedule individual sensor nodes into the sleep state in which they can maximize their energy conservations. Efficient sleep scheduling protocols [2, 7, 8, 11, 15, 17] have been extensively studied. In [2], the author proposes an efficient sleeping scheduling algorithm based on application time, which aggressively exploits the timing semantics of WSN applications. The essential algorithm consists of two parts: a traffic shaper and a local scheduling algorithm. In [7], the author considers designing an efficient wakeup scheduling algorithm while adhering to the bidirectional end-to-end delay constraints posed by certain applications. An energy-efficient and low-latency MAC (DMAC) is proposed in [8], which is optimized for data gathering trees in WSNs. The paper also addresses the data forwarding interruption problem, which is solved by employing a staggered active/sleep schedules. A flexible MAC protocol is proposed in [17]. The rationale behind is the introduction of the synchronized periodic duty cycling of sensor nodes so that the costs on idle listening can be reduced. However, their

scheduling capabilities are limited due to their intrinsic characteristics, such as a lack of explicit interactions with the application layer modules.

In this paper, we consider an important overarching problem that encompasses both aspects of WSN power management. A new Data-Driven Power Management framework (DDPM) is proposed which integrates two techniques: approximate querying and sleep scheduling. By aggressively and judiciously exploiting beneficial properties from these two techniques, we can achieve better energy efficiency. The rationale behind the DDPM is that the trajectories of the sensor values are exploited when constructing the nodal level sleep scheduling. Consequently, it leads to a more flexible and efficient power management mechanism.

The DDPM is comprised of three main modules namely Approximate Query Management (AQM), Interval Predictor (IP), and Sleep Scheduler (SS). The AQM module manages all the error tolerances that are assigned to a node. The IP module explicitly interacts with the AQM making predictions based on these locally stored error bounds. Each prediction is represented by a time span. It indicates the maximum length of time that the sensed value of a node takes to reach a particular error bound. We adopted a Kalman Filter based approach [5, 6] as the core of the IP module. Eventually, the SS module can determine the sleep schedule for each sensor node based on these predictions.

The design of the DDPM scales well with the number of nodes since it requires only locally available information, such as nodal error bounds and sensor values. Nevertheless, the independent nature of the DDPM can cause significant communication delays along the multi-hop structure of the WSN (e.g. staggered schedules). In order to mitigate the problem, a coordination component is integrated with the SS module. It eventually exploits the tradeoffs between the nodes' sleep times and the end-to-end communication latency so that different quality attributes of the WSN can be achieved.

The rest of the paper is organized as follows. In section II, we describe the system model, and formally define the problem. Section III presents the holistic view of the DDPM along with the detailed designs of each module. Section IV provides extensive experimental studies on the DDPM, with section V concluding the paper.

II. SYSTEM MODELING AND PROBLEM FORMULATION

A. Network Model

We consider a large-scale WSN that consists of numerous sensor nodes and a base station. These nodes are distributed into a physical area self-organized into a connected network. They periodically collect in-situ data, such as light and humidity. The collected data are sent back to the base station either directly or via several relay nodes. For those nodes that can not directly reach the base station, each of them can choose one relay node from a number of potential candidates. From a topology point of view, we consider a hierarchical network structure. Each node is assigned with a virtual level number based on its hop-distance to the base station. Nodes with bigger level numbers are further away from the base station than those nodes with smaller level numbers. A node is regarded as the CHild node (*CH*) of its immediate relaying node, and the relaying node is regarded as the PArent node (*PA*).

Identify applicable sponsor/s here. *(sponsors)*

B. Communication Model

Two types of links are considered between two nodes: Reachable Link (*RL*) and Communication Link (*CL*). A *RL* between two nodes indicates that they are able to communicate with each other. We consider that a *CH* can have multiple *RL*s with different *PA*s. On the other hand, the *CL* is the actual communication link that is strategically chosen by the *CH* for the actual data transmission. It is possible for a *CH* to transmit its data to different *PA*s at the same time (e.g. broadcast or multicast), and by doing so the reliability can be enhanced. Nevertheless, it also increases the energy consumption for transmitting redundant data. Hence, we consider that each *CH* can only have one *CL* for its data transmission. Each *RL* of a *CH* serves as a potential candidate for becoming a *CL*.

C. Query Model

Continuous approximate queries are considered in this paper. A query is composed of the following key elements:

<{Attribute}><Range><Time Window><Interval><Error>

where *{Attribute}* defines a set of application specific interests, and *Range* is used for defining the size of the queried area. *Time Window* rules how long a query is valid for, whereas *Interval* sets the temporal granularity. These four elements are typical in WSN DBMS, such as TinyDB [9] and Coguar [16]. Besides these attributes, an approximate query carries an extra *Error* element, which specifies the error tolerance that the application can bear. Each queried node i can then be allocated with a local error tolerance e_i. The value then can be used to determine whether a node needs to send its data back to the base station or not. For example, given a time instance T_0, we assume that the sensed value of node i at T_0 is $V_0(i)$. Without installing the e_i bound, at T_1 node i needs to send its sensed value $V_1(i)$ to the base station. However, if an e_i is installed at i, then node i sends $V_1(i)$ only if the value falls outside the range $[V_0(i) - e_i/2, V_0(i) + e_i/2]$. Otherwise, the transmission can be suppressed. By exploiting the temporal correlations among sensed data, the energy can eventually be saved on sending unbiased values.

D. Energy Model

We consider that there is no energy constraint at the base station since it normally has abundant resources. On the other hand, we consider that sensor nodes are very energy-constrained because they are normally battery-powered. Among different node operations, communication is the most expensive operation with idling being the second. Hence, the node energy consumption can be significantly saved by reducing the communication activities of a sensor node and switching it off when it becomes idle.

E. Problem Definition

In this paper, we consider the combination of two energy efficient techniques: the approximate querying and the sleep

scheduling. By integrating them into one framework, we can achieve better energy efficiency while still meeting different application specific criteria, such as data accuracy and communication latency. The essential problem is how can we efficiently exploit the collaboration between these two techniques? In this section, we formally describe the underlying problem.

Consider an approximate query with an error tolerance value E, it is disseminated to N sensor nodes. Each node n_i is allocated with a local error tolerance value $e(n_i)$. In order to satisfy the precision constraint of the approximate query, the condition below should be met:

$$\sum_{i=1}^{N} e(n_i) \le E \tag{1}$$

The optimal allocation of $e(n_i)$ to each node is an independent problem, which has been extensively studied in the literature [15]. To simplify the problem presentation, we adopt a uniform error tolerance allocation model, which is shown below:

$$\forall i : e(n_i) = \frac{E}{N}, \quad 1 \le i \le N \tag{2}$$

By taking advantage of this tolerance value $e(n_i)$, each node can make a prediction on how long it can safely sleep until any of its sensed values violate the assigned precision constraint. This can be achieved by installing an Interval Predictor (IP) module on each node. We defer the discussion on the IP to section III-B. Here we assume that the IP has a prediction function P, which takes $e(n_i)$ as its input and generates a value $P(e(n_i))$. This value indicates the predicted maximum sleep time $S_{max}(n_i)$ of a node n_i while sensed values of n_i during that period are still bounded with its error tolerance $e(n_i)$. Hence, we can formally derive the objective as follows:

$$\max: \quad \sum_{i=1}^{N} S_{\max}(n_i) \tag{3}$$

So far we just consider the local behavior of the network (e.g. maximizing the local sleep times for in-network nodes based on their locally available information). It is worth noting that these local behaviors can lead to significant network-wide communication delays. The side effect is due to the fact that once a node enters into the sleep state, all the communication via this node is postponed until it wakes up again. Due to the dynamic characteristics of the WSN, it is possible that these predicted nodal level $P(e(n_i))$s are all staggered. Considering a multi-level routing structure, uncoordinated scheduling can result in significant latency for lower-level nodes sending updates back to the base station. In the worst situation, the benefits from making a prediction on $P(e(n_i))$ is evened out. In addition, for time-critical WSN applications any extra latency can not be tolerated.

In this paper, we address the problem of maximizing the length of sleep time for individual nodes while avoiding the undesirable communication delays (MaxSMinL). Formally, the WSN is modeled as a undirected graph $G(N, E)$ where N is the set of nodes including the base station, and E is the set of RLs. The base station is denoted as n_0. We represent the network topology as an incidence matrix $M \in \{0, 1\}^{|N|*|N|}$. We define each element m_{ij} of M as:

$$m_{ij} = \begin{cases} 1, & \text{if there is a } RL \text{ between node } i \text{ and } j \\ 0, & \text{otherwise} \end{cases}$$

We assume the existence of a Transformation Function (TF). The TF takes an M as its input, and generates an $M'_{|N|*|N|}$ as the output. The element m'_{ij} of M' is denoted as:

$$m'_{ij} = \begin{cases} 1, & \text{if there is a } CL \text{ between node } i \text{ and } j \\ 0, & \text{otherwise} \end{cases}$$

Eventually, the underlying problem can be formally described as follows:

Maximize

$$\sum_{i=0}^{|N|} \sum_{j=0}^{|N|} Syn(S_{\max}(i), S_{\max}(j)) \times m'_{ij} \tag{4}$$

Minimize

$$\frac{\sum_{i=0}^{|N|} \sum_{j=0}^{|N|} L(i,j) \times m'_{ij}}{|Deg|} \tag{5}$$

Subject to

$$\sum_{i=0}^{|V|} \sum_{j=0}^{|V|} m'_{ij} = 2 \times (|N| - 1) \tag{6}$$

$$\forall i : S_{\max}(i) \ge 1, \qquad 1 \le i \le |N| \tag{7}$$

where the TF function can have different objectives by employing different strategies, we give more detailed discussions on these strategies in section III. The $Syn(i, j)$ determines the maximum sleep times of both nodes i and j while still keeping them synchronized. The $L(i, j)$ is the latency function measuring the communication delay between nodes i and j. The output calculation of the given functions: Syn and L are be provided in a later section. $|Deg|$ is the out-degree of the base station, which is used to normalize the latency for each path linking to the base station. Equation (4) describes the objective function of the MaxSMinL which is to maximize the overall sleep time for all nodes. In order to simplify the presentation, we consider minimizing the average latency of the WSN as the second objective which is described in equation (5). The first constraint is that each node can only be assigned to one PA node (one CL per node) except for the base station, which is described by equation (6). The predicted S_{max} value of any node is lower-bounded by the second constraint (equation (7)); if it equals to one, it indicates that the sleep time of a node is minimized.

The MaxSMinL problem belongs to the multi-objective optimization, which is inherently NP-Hard. Therefore, we convert the original MaxSMinL to a new problem, named MaxS, with a single objective. One of the objectives, equation (5), is changed to a constraint as follows:

$$\frac{\sum_{i=0}^{|N|} \sum_{j=0}^{|N|} L(i,j) \times m'_{ij}}{|Deg|} \le \varepsilon, \varepsilon \ge 0 \tag{8}$$

where ε is a user-defined value for tuning the balance between the overall sleep time and the latency. It is obvious that a solution exists if ε is big enough. We can show that there are feasible solutions even if $\varepsilon = 0$ through an auxiliary example.

Example. Figure 1 shows a three-level WSN with one base station node n_0, and seven sensor nodes (n_1 to n_7). If two nodes can directly communicate with each other, then there is an edge between them. Accordingly we can categorize these nodes either as *PA* nodes or *CH* nodes. In particular, we can view n_1 to n_3 as PA_1 to PA_3 and n_4 to n_7 as CH_4 to CH_7. We assume that the base station node is available all the times since it has relatively abundant resources. All non-BS nodes are available according to their S_{max} values. For calculating the latency between two nodes (function *L*), we examine the congruence relationship between their S_{max} values. We consider zero latency when two S_{max} values are congruent providing the *PA* wakes up more frequently than its *CH*. Otherwise, the latency is their Greatest Common Divisor (GCD) minus the S_{max} value of the *CH*. The equation (7) constrains that each *CH* can only choose one *RL* (dotted lines) as the *CL* (solid lines) for relaying its transmission. Having constructed such a network, now we show that by tuning the value of ε different objectives can be achieved.

Maximize sleep time: The overall network sleep time is upper-bounded by $\sum S_{max}(n_i)$ where $1 \le i \le 7$. By slacking the latency constraint, we can achieve the maximized overall sleep time as *23* units of time in this example. Eventually we can derive the lower-bound of ε as *5*. This demonstrates that the optimal solution can be found with respect to a satisfactory ε value.

Minimize latency: By exploiting the tradeoff between the sleep time and the latency, we can achieve minimized average latency as *0*. However, the child node CH_6 needs to sacrifice one unit of its S_{max} making the overall sleep time *22*. This demonstrates that there exists a feasible solution (also close to the optimal solution) while the average latency is minimized.

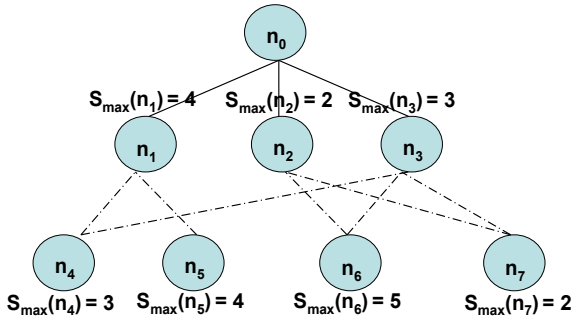

Figure 1. An auxiliary WSN model

III. DATA-DRIVEN POWER MANAGEMENT FRAMEWORK

A Data-Driven Power Management Framework (DDPM) is proposed to fully exploit the energy-saving opportunities for sensor nodes while they can still meet the application-specific requirements. The rationale behind the DDPM is that it combines and coordinates the approximate querying and the sleep scheduling techniques to further improve the energy-efficiency. Compared with the existing approximate querying techniques [4, 12], the DDPM takes a further step by predicting the maximum length of times that a node can be safely turned into the sleep state without violating its data accuracy. In this section, we first start with the holistic view of the DDPM, and then detailed description on the design of each module is presented.

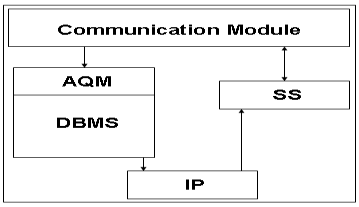

Figure 2. DDPM

A. DDPM – A Holistic View

The DDPM is installed at each sensor node that participates in nodal activities. Figure 2 shows a schematic view of the DDPM framework and its components. Basically the DDPM consists of the following main modules:

Approximate Query Manager (AQM): the AQM serves as the interface between the approximate queries and the node DBMS. Meanwhile, it also stores error tolerance values for all running queries. Mainly it is responsible for providing efficient query services.

Interval Predictor (IP): the IP module collaboratively works with the AQM. It enquires the AQM with the allocated error tolerance value for a particular query, and then it estimates the maximum length of times the node can sleep while its sensed value still stays inside the tolerated range. By taking this further step, the IP can help the node to achieve better energy efficiency while still meeting the data accuracy requirements of the application.

Sleep Scheduler (SS): once the IP generates a prediction, the result is passed onto the SS module. The SS utilizes this result to schedule the sleep time and the wakeup time for each sensor node.

B. Interval Predictor (IP)

The purpose of integrating an IP is that it can periodically generate estimations on how much time a node can be safely switched into sleep state. Therefore, the capability of the IP largely relies on the underlying prediction technique it adopts.

Intensive researches have been dedicated to develop efficient and reliable prediction-based techniques for the WSN applications. Some of pioneering work can be found in [3, 5, 6]. They all share a common objective which is to capture the trajectories of the sensed data by judiciously exploiting spatial and temporal correlation among them. These trajectories then can be used as the bases for inferring other values. Hence, sensor nodes only need to send their results when these results are biased from the predicted values, so eventually energy can be saved on sending unbiased ones.

In this paper, we adopted a Kalman Filter (KF) based approach as the underlying prediction technique of the IP. It was well studied in [5, 6] as an estimation technique to adaptively adjust the sampling rate of the sensor node within a given range. Compared with other techniques, it is very light-weighted which makes it especially suitable for energy-

constrained environment. Moreover, it is capable of generating satisfactory results even provided with inaccurate information.

Recent work in [10] has shown improvements on the KF based approach in terms of energy reduction and estimation accuracy. Since the focus of the paper is on the integration of different energy-efficient techniques, we draw the line for further discussions on them. However, a better prediction technique is still complementary to our framework.

C. Sleep Scheduler (SS)

The SS module plays an important role in the whole framework. The main responsibility of a SS is to make sleep schedule for each node that can maximize their sleep times while still meeting the application-specific constraints. The SS starts with analyzing the estimation generated by the IP, and then it determines how long a particular node can be scheduled for sleep. Due to the underlying local behaviors of the IP, this can lead to staggered schedules along the routing path. Globally, the asynchronous coordination among different nodes might potentially cause significant delays. On the other hand, global scheduling is too expensive and does not scale very well with the increasing size of the WSN. Hence, it is vital to design an efficient SS which can complement the integration of the IP and the approximate querying technique. The challenging issue lies ahead is the design of a decentralized scheme for the efficient coordination among different nodal level schedules, so that the sleep times of the in-network nodes are maximized while the average communication latency is satisfactory.

Algorithm STM:
1: *Node s gets the sleep times of its direct neighbors* $N = \{N_i | 1 \leq i \leq n\}$
2: $min = L(N_1, s)$ // *initialize the minimum latency*
3: $Relay(s) = N_1$ // set relay node of s to N_1
4: FOR $j = 2$ to n
5: IF $L(N_j, s) \leq min$ THEN
6: $min = L(N_j, s)$
7: $Relay(s) = N_i$
8: END FOR

Figure 3. Pseudo code of STM

1) Coordination and Optimization: Since it is intractable to consider both maximizing the sleep time and minimizing the latency at the same time, we instead propose several heuristic algorithms for exploiting different tradeoffs between them. These algorithms also imply different strategies for implementing the *TF* function. In order to address the scalability issue, the proposed algorithms only utilize locally available information.

a) Sleep Time Maximization (STM): We assume that each node has knowledge about its direct neighbors. Without changing the length of the nodal sleep time, the algorithm allows each node to choose its relay node based on minimizing the latency function *L*. If there is a tie, the node with smaller ID can be chosen. The details of the algorithm are shown in Figure 3.

b) Pairwise-based Algorithm (PWA): The drawback of the *STM* is its high communication latency, which makes it not suitable for these time-critical WSN applications. On the other hand, we can observe the possibility of sacrificing some nodal level sleep times for reducing the latency from the auxiliary WSN (Figure 2). However, it is prohibitively expensive to tailor each node's sleep time for finding a coordination plan with the optimized latency (especially for a multi-hop WSN). Alternatively, we propose the *PWA* for mitigating the problem.

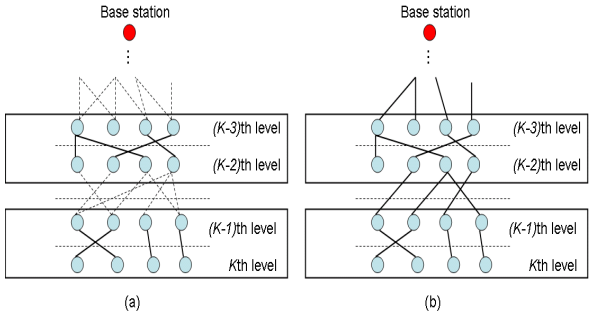

Figure 4. An ancillary example of PWA execution. (a) result of PS. (b) result of BS.

The *PWA* embodies two stages called Pairing Stage (PS) and Bridging Stage (BS). During the PS, levels is formed into a number of pairs in light of user pre-defined rules. For example, let *k* denote the furthermost level to the base station, then it can pair with *(k-1)th* level, and iteratively *(k-2)th* level can form pair with *(k-3)th* level. It is possible that level *1* is singled out, e.g. if *k* is odd. If this is the case, then level *1* can skip the PS. Once all levels successfully pair up with each other, we can start tunning the tradeoff between nodal sleep times and the latency based on these pair structures. The final result of the PS is shown in Figure 4.(a), where each *CH* of a pair reinforces one of its *RL* as the *CL* so that it can meet the user or application specified requirement (e.g. zero latency). Meanwhile, the sleep time of each node can also be adjusted accordingly. After the PS, the WSN enters into the BS, during which stage different pairs are connected with each other to form the final network wide coordination plan (Figure 4.(b)). The algorithmic part for achieving this is similar to the *STM*. The main idea is to avoid the ripple effect of changing resultant intra-pair coordination plan from the PS.

There exist different strategies for the calculation of the *Syn* function, which is inherently related to the process of making intra-pair coordination plan. On the other hand, different child-parent relationship models can also lead to different strategies for making the plan. Based on these observations, we first identify all the models describing the child-parent relationship, and then relative optimizations on calculating the result of *Syn* are discussed. Due to the space limit, we only consider the zero-latency requirement while calculating the optimized *Syn*.

Model: One parent -> one child. This is a simplex relation with two nodes: one CH_j and one PA_i. Given two error tolerance values $e(CH_j)$ and $e(PA_i)$, their predicted maximum length of sleep times are $S_{max}(CH_j)$ and $S_{max}(PA_i)$. The ideal (maximum) schedules for both nodes can be their originally predicted maximum sleep times, but there are some cases that the optimum can not be achieved owing to the zero-latency constraint. In these cases, in order to maintain the zero-latency

constraint, S_{max} value has to be compromised by shrinking either node CH_j or PA_i since decreasing S_{max} value does not breach the accuracy requirement either. Two cases can be enumerated to describe the difference λ between the given $S_{max}(CH_j)$ and $S_{max}(PA_i)$:

$$\lambda = S_{\max}(PA_i) - S_{\max}(CH_j)\begin{cases}\le 0\\ > 0\end{cases}$$

If λ is less than *0*, that means the updating rate of PA_i is more frequent than the updating rate of CH_j, nevertheless it is still possible that extra latency can be incurred if their schedules are staggered. Under the worst situation, the delay $L(CH_j, PA_i)$ could be as much as $S_{max}(CH_j)$ * $S_{max}(PA_i)$ - $S_{max}(CH_j)$. In order to avoid the undesirable delay, we further apply the modulo operation on $S_{max}(CH_j)$ and $S_{max}(PA_i)$, and then the sum of the optimal synchronized sleep schedule $S_{Opt}(S_{max}(PA_i), S_{max}(CH_j))$ for both nodes can be derived as follows:

$$\begin{aligned}&S_{Opt}(S_{\max}(PA_i), S_{\max}(CH_j)) =\\ &S_{\max}(PA_i) + S_{\max}(CH_j) - Mod(S_{\max}(CH_j), S_{\max}(PA_i))\end{aligned} \quad (9)$$

where *Mod(i, j)* is the modulo function which returns the remainder of dividing *i* by *j*. This function essentially checks the congruence relationship between two sleep schedules. If two schedules are not congruent, then $S_{max}(CH_j)$ shrinks accordingly and only sub-optimal sleep time schedules can be achieved. On the other hand, even if two schedules are staggered, the optimal sleep schedules can still be achieved providing their congruence relationship is true (e.g. *Mod* function returns *0*).

If λ equals to *0*, two nodes are fully synchronized. No shrinking action is taken place since the zero-latency constraint can be satisfied. The optimal value for $S_{Opt}(S_{max}(PA_i), S_{max}(CH_j))$ is:

$$\begin{aligned}&S_{Opt}(S_{\max}(PA_i), S_{\max}(CH_j)) =\\ &S_{\max}(PA_i) + S_{\max}(CH_j) = 2\times S_{\max}(CH_j) = 2\times S_{\max}(PA_i))\end{aligned} \quad (10)$$

If λ is greater than *0*, this is the case that the updating rate of the *CH* is faster than the updating rate of its *PA*. Under this one-to-one relationship, there is not much optimization can be done. In order to cope with the communication needs of its *CH*, the *PA* changes its $S_{max}(PA_i)$ to $S_{max}(CH_j)$. Eventually, the optimal value for $S_{Opt}(S_{max}(PA_i), S_{max}(CH_j))$ is:

$$S_{Opt}(S_{\max}(PA_i), S_{\max}(CH_j)) = 2\times S_{\max}(CH_j) \quad (11)$$

Model: One parent -> many children. This is a more complex situation but still a very common one in the WSN: one PA_i and a set of *m* child nodes $CH = \{ch_j | 1 \le j \le m\}$. Consider the predicted maximum length of sleep time for PA_i is $S_{max}(PA_i)$, and $S_{max}(ch_j)$ is the predicted sleep time of the *jth* node of the *CH*. With more child nodes, the chances for forming staggering schedules also increase. The worst case is that all the predicted sleep times of *CH*s are smaller than the sleep time of their parent node PA_i. Hence, we shall first make sure that the most stringent communication needs of *CH* can be met by PA_i. Suppose that a child node ch_{min} of *CH* has the shortest length of predicted sleep time $S_{max}(ch_{min})$, where $1 \le min \le m$. We then first check the λ value defined as follows:

$$\lambda = S_{\max}(PA_i) - S_{\max}(ch_{\min})\begin{cases}\le 0\\ > 0\end{cases}$$

If $\lambda \le 0$, it indicates the updating rate of the parent node PA_i is no slower than the updating rate of any of its child node *CH* $S_{max}(PA_i) \le S_{max}(ch_{min})$. Hence, we can derive the $S_{Opt}(S_{max}(PA_i), S_{max}(CH))$ value as:

$$\begin{aligned}&S_{Opt}(S_{\max}(PA_i), S_{\max}(CH)) =\\ &S_{\max}(PA_i) + \sum_{j=1}^{m}(S_{\max}(ch_j) - Mod(S_{\max}(ch_j), S_{\max}(PA_i)))\end{aligned} \quad (12)$$

Based on the above equation, we can observe that if and only if the accumulated values of *Mod* function equals to *0*, the optimal value can be achieved. Otherwise, schedules of some child nodes need to be shrunken by the corresponding remainder of the *Mod* function in (12).

If $\lambda > 0$, it implies that we first have to shrink the sleep time of the PA_i so that we can avoid incurring any extra delay. The new S_{max} of PA_i can be the same as $S_{max}(ch_{min})$ so that the PA_i can cope with the communication needs of its most stringent child node ch_{min}. After the parent node adjusts its schedule, all its child nodes can also make adjustments accordingly. Eventually the value of $S_{Opt}(S_{max}(PA_i), S_{max}(CH))$ can be obtained as below:

$$S_{ajust}(ch_j) = S_{\max}(ch_j) - Mod(S_{\max}(ch_j), S_{\max}(ch_{\min})) \quad (13)$$

$$S_{Opt}(S_{\max}(PA_i), S_{\max}(CH)) = S_{\max}(ch_{\min}) + \sum_{j=1}^{m} S_{adjut}(ch_j) \quad (14)$$

The worst situation ($\lambda < 0$) causes the consequence that all the child nodes are forced to reduce their sleep times to some extent. The best case is that the accumulated value of the Mod function equals to *0*.

Intuitively the child node should prefer to join these parent nodes that can help it achieve higher S_{Opt} values. However, sometimes judgment based on S_{Opt} only can be misleading. Consider an example that a child node CH_1 has two *RL*s with PA_1 and PA_2. Their predicted sleep times are:

$$S_{\max}(PA_1) = 4$$
$$S_{\max}(PA_2) = 3$$
$$S_{\max}(CH_1) = 2$$

It is the case $\lambda > 0$ with the one-to-one model. The sleep time of PA_1 and PA_2 must be reduced to 2 units as below:

$$S_{Opt}(S_{\max}(PA_1), S_{\max}(CH_1)) = 2 * S_{\max}(CH_1) = 4$$
$$S_{Opt}(S_{\max}(PA_2), S_{\max}(CH_1)) = 2 * S_{\max}(CH_1) = 4$$

Suppose that child node CH_1 has only two parents, either PA_1 or PA_2 can be selected. Nevertheless, PA_1 sacrifices half of its sleep times for being the parent of CH_1, whereas PA_2 loses only one third. In order to resolve this issue, we introduce an approximation variable α which evaluates the ratio between the S_{Opt} of two nodes and the sum of their originally predicted sleep

times. Given the above example, two α values can be derived as follows:

$$\alpha_1 = \frac{S_{Opt}(S_{\max}(PA_1), S_{\max}(CH_1))}{S_{\max}(PA_1) + S_{\max}(CH_1)} = \frac{2+2}{4+2} \approx 66.67\%$$

$$\alpha_2 = \frac{S_{Opt}(S_{\max}(PA_2), S_{\max}(CH_1))}{S_{\max}(PA_2) + S_{\max}(CH_1)} = \frac{2+2}{3+2} \approx 80\%$$

The essential of the approximation variable is to capture how much of the originally predicted sleep times can be preserved after the synchronization. Hence, locally each child node can choose its parent node by picking the one with the highest α value. The node ID is used to break the tie if there are multiple parent nodes having the same highest α value with a child node.

IV. PERFORMANCE EVALUATION

In this section, two algorithms, namely the STM and the PWA, are simulated and compared. We study how average latency and overall sleep time are affected by varying number of nodes and network topologies. Additionally, we adopted three probability distribution models to mimic the behaviors of the IP module, which assigns a sleep time to each node. These models are Normal Distribution, Uniform Distribution, and Zipf's Distribution. In order to simplify the presentation while still keeping the generality, we set the range of the IP prediction as *2-5*. We also assume a fully connected network. For a particular experiment setting, we run it repeatedly for *20* times in order to get the mean values and standard deviations as well.

A. Average Latency

Figure 5 shows the mean average latency versus the total number of sensor nodes. In this experiment, we vary the number of sensor nodes while fixing the network topology (in terms of levels). Then each node is randomly assigned to a level from *1* (closest to the base station) to *5* (farthermost to the base station). We can observe that with the increasing number of sensor nodes, the mean latencies derived from both algorithms are decreasing. Since more sensor nodes enable more options for each node to choose its relay node. Therefore, deploying more sensor nodes is likely to improve the latency. Nevertheless, the PWA outperforms the STM under all three distribution models while there are a relatively small number of nodes (e.g. *100*). The Uniform Distribution disperses the sleep time equally to each node, so the STM incurs relatively large latency. However, the PWA still performs very stably regardless the model used.

Figure 6 illustrates the mean latency versus the number of levels (various network topologies). In this experiment, we fix the number of nodes as *100* while changing the total levels of the network from *5* to *10*. The STM acts more drastically to the changes of the network topology. Generally, the STM incurs more communication latencies when the network becomes deeper from the hierarchy point of view. On the other hand, the PWA is almost immune to the network topology changes. However, the incurred mean latencies under Normal Distribution model are more irregular. Though with less levels, the derived mean latencies from the STM is close to the ones derived from the PWA, the PWA constantly outperforms the STM when the network is divided into more levels.

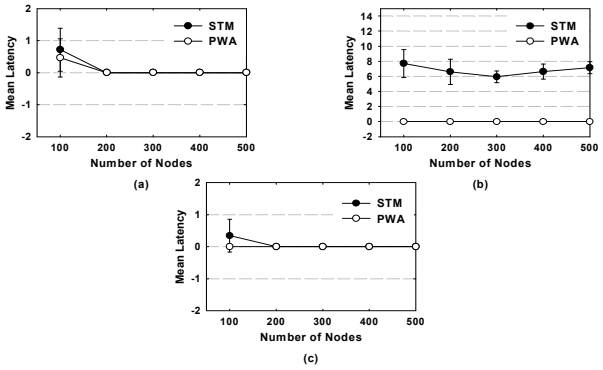

Figure 5. Mean latency under varying number of nodes. (a) Normal. (b) Uniform. (c) Zipf.

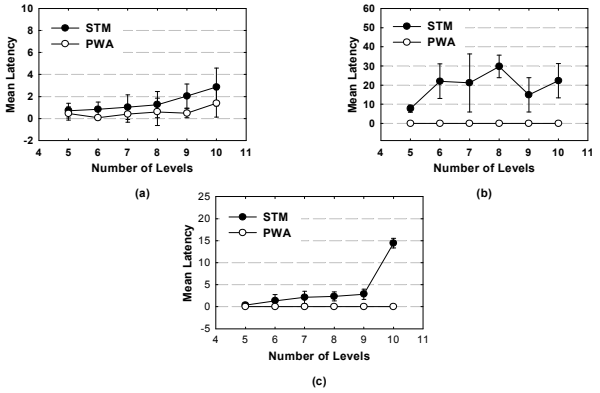

Figure 6. Mean latency under varying network topologies. (a) Normal. (b) Uniform. (c) Zipf.

B. Overall Sleep Time

Figure 7 plots the mean overall sleep time versus the number of nodes. In this experiment, we also fix the network topology while varying the total number of in-network nodes. Sleep time of each sensor node is added together to form the overall sleep time. Since the STM does not utilize the shrinking technique, it can achieve maximized overall sleep time. The results derived from the STM can serve as the benchmark for evaluating the energy-efficiency of the PWA. We can observe that the PWA can effectively reduce the communication latency while maintaining very close to optimal sleep times.

Figure 8 displays the mean overall sleep time versus the varying network topologies. In this experiment, the total number of sensor node is fixed as *100* while changing the total levels of the WSN. As we can see, the performance of the PWA is still very promising in spite of the network topology changes and different distribution models. It is worth noting that the communication latency increases with the rising number of network levels. Hence, in order to achieve shorter delay, more nodes need to sacrifice their sleep times. Since the

PWA utilizes a pair wise based approach, it can limit the impact of shrinking nodal sleep time on the overall network energy-efficiency. We can still observe very close to optimal results from this experiment.

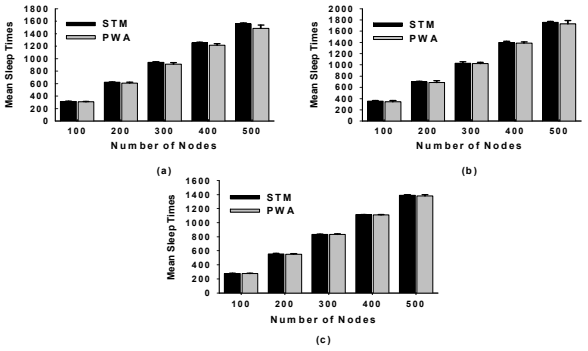

Figure 7. Mean overall sleep time under varying number of nodes. (a) Normal. (b) Uniform. (c) Zipf.

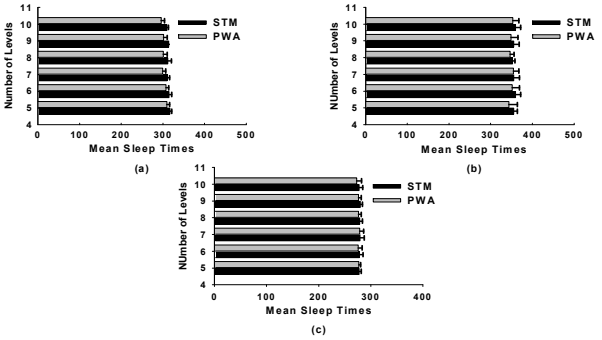

Figure 8. Mean overall sleep time under varying network topologies. (a) Normal. (b) Uniform. (c) Zipf.

V. CONCLUSION

We proposed a novel and energy efficient DDPM to exploit the beneficial properties from the approximate querying and the power management techniques. We have also addressed the important problem of maximizing the overall WSN sleep time while minimizing the incurred communication delay. The problem was first formulated as a multi-objective optimization, which is NP-hard. We then proposed two heuristic algorithms (STM and PWA) for achieving different objectives, such as sleep time maximization and latency optimization with sleep time tradeoffs. Experimentally we demonstrated that the proposed DDPM could significantly increase the overall sleep time while incurring only a little communication latency.

REFERENCES

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A Survey on sensor networks," IEEE Communications Magazine, vol. 40(8), pp. 102-114, 2002.

[2] O. Chipara, C. Lu, and G.-C. Roman, "Efficient Power Management based On Application Timing Semantics for Wireless Sensor Networks," presented at the 25th IEEE International Conference on Distributed Computing Systems, Columbus, Ohio, USA, 2005.

[3] S. Goel and T. Imielinski, "Prediction-based Monitoring in Sensor Networks: Taking Lessons from MPEG," ACM Computer Communication Review, vol. 31, 2001.

[4] Q. Han, S. Mehrotra, and N. Venkatasubramanian, "Energy Efficient Data Collection in Distributed Sensor Environments," in the 24th International Conference on Distributed Computing Systems. Tokyo, Japan, 2004.

[5] A. Jain and E. Y. Chang, "Adaptive Sampling for Sensor Networks," in the First Workshop on Data Management for Sensor Networks (DMSN 2004): in conjunction with VLDB 2004. Toronto, Canada, 2004.

[6] A. Jain, E. Y. Chang, and Y.-F. Wang, "Adaptive stream resource management using Kalman Filters," in the ACM SIGMOD/PODS conference (SIGMOD '04). Paris, France, 2004.

[7] A. Keshavarzian, H. Lee, and L. Venkatraman, "Wakeup Scheduling in Wireless Sensor Networks," in the 7th ACM International Symposium on Mobile Ad Hoc Networking and Computing. Florence, Italy, 2006.

[8] G. Lu, B. Krishnamachari, and C. S. Raghavendra, "An adaptive energy-efficient and low-latency MAC for data gethering in wireless sensor networks," presented at 18th International Parallel and Distributed Processing Symposium, 2004, Santa Fe, New Mexico, USA, 2004.

[9] S. R. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong, "TinyDB: an acquisitional query processing system for sensor networks," ACM Transactions on Database Systems (TODS), vol. 30, pp. 122 - 173, 2005.

[10] S. Santini and K. Romer, "An Adaptive Strategy for Quality-Based Data Reduction in Wireless Sensor Networks," in Third International Conference on Networked Sensing Systems. Chicago, Illinois, USA, 2006.

[11] M. L. Sichitiu, "Cross-layer scheduling for power efficiency in wireless sensor networks," presented at Twenty-third Annual Joint Conference of the IEEE Computer and Communications Societies, INFOCOM 2004, Hong Kong, China, 2004.

[12] A. Silberstein, R. Braynard, and J. Yang, "Constraint Chaining: On Energy-Efficient Continuous Monitoring in Sensor Networks," in 2006 ACM SIGMOD International Conference on Management of Data. Chicago, Illinois, 2006.

[13] M. Tang and J. Cao, "Optimization on Distributed User Management in Wireless Sensor Networks", presented at ICPADS 2007, Hsinchu, Taiwan, 2007

[14] X. Tang and J. Xu, "Extending Network Lifetime for Precision-Constrained Data Aggregation in Wireless Sensor Networks," in the 25th Conference on Computer Communications, Joint Conference of the IEEE Computer and Communications Societies. Barcelona, Catalunya, Spain, 2006.

[15] H. Wu, Q. Luo, and W. Xue, "Distributed cross-layer scheduling for in-network sensor query processing," presented at Fourth Annual IEEE International Conference on Pervasive Computing and Communications, PerCom 2006, Pisa, Italy, 2006.

[16] Y. Yao and J. Gehrke, "The cougar approach to in-network query processing in sensor networks," ACM SIGMOD Record, vol. 31, pp. 9 - 18, 2002.

[17] W. Ye, J. Heidemann, and D. Estrin, "An energy-efficient MAC protocol for Wireless Sensor Networks," in the 23rd Annual Joint Conference of the IEEE Computer and Communications Societies. Hong Kong, China: IEEE, 2002.