Optimal Anycast Technique for Delay-Sensitive Energy-Constrained Asynchronous Sensor Networks

Joohwan Kim, *Student Member, IEEE*, Xiaojun Lin, *Member, IEEE*, and Ness B. Shroff, *Fellow, IEEE*

***Abstract*—In wireless sensor networks, asynchronous sleep-wake scheduling protocols can be used to significantly reduce energy consumption without incurring the communication overhead for clock synchronization needed for synchronous sleep-wake scheduling protocols. However, these savings could come at a significant cost in delay performance. Recently, researchers have attempted to exploit the inherent broadcast nature of the wireless medium to reduce this delay with virtually no additional energy cost. These schemes are called "anycasting," where each sensor node forwards the packet to the first node that wakes up among a set of candidate next-hop nodes. In this paper, we develop a delay-optimal anycasting scheme under periodic sleep-wake patterns. Our solution is computationally simple and fully distributed. Further, we show that periodic sleep-wake patterns result in the smallest delay among all wake-up patterns under given energy constraints. Simulation results illustrate the benefit of our proposed schemes over the state-of-the art.**

***Index Terms*—Anycast, Sleep-wake scheduling, Sensor network, Energy-efficiency, Delay, Periodic wake-up process**

I. Introduction

The most efficient method to save energy in wireless sensor networks (WSNs) is to put nodes to sleep when there is no need to relay or transmit packets. Such mechanisms are called *sleep-wake scheduling* and have been used to dramatically reduce energy consumption in energy-constrained WSNs. However, it is well known that sleep-wake scheduling can significantly increase the packet-delivery delay because, at each hop, an event-reporting packet has to wait for its next-hop node to wake up. Such additional delays can be detrimental to delay-sensitive applications, such as Tsunami/fire detection, environmental monitoring, security surveillance, etc. In this paper, we study how to improve this tradeoff between energy-savings and delay, by using a technique called "*anycasting*" (to be described later) that exploits the broadcast nature of the wireless medium.

In the literature, many synchronous sleep-wake scheduling protocols have been proposed [2]–[6]. In these protocols, sensor nodes periodically exchange synchronization messages with neighboring nodes. However, this message exchange inevitably incurs additional communication overhead, and consumes a considerable amount of energy. In this paper, we focus on asynchronous sleep-wake scheduling, where nodes do not synchronize their clocks with other nodes and thus wake up independently [7]–[9]. Asynchronous sleep-wake scheduling is simpler to implement, and it does not consume energy required for synchronizing sleep-wake schedules across the network. However, because nodes do not know the wake-up schedules of other nodes, they have to estimate the wake-up schedule, which can result in additional delays that could detrimental to delay-sensitive applications.

Recently, *anycast packet-forwarding schemes* have been shown to substantially reduce the one-hop delay under asynchronous sleep-wake scheduling [10]–[20]. Note that in traditional packet-forwarding schemes, nodes forward packets to their designated next-hop nodes. In contrast, in anycast-based forwarding schemes, nodes maintain multiple candidates of next-hop nodes and forward packets to the *first* candidate node that wakes up. Hence, an anycast forwarding scheme can substantially reduce the one-hop delay over traditional schemes, especially when nodes are densely deployed, as is the case for many WSN applications. (See the example in Section I and Fig. 1 of [20] that illustrates the advantage of anycasting over traditional schemes.) However, the reduction in the one-hop delay may not necessarily lead to a reduction in the expected end-to-end delay experienced by a packet because the first candidate node that wakes up may not have a small expected end-to-end delay to the sink. Hence, the anycast forwarding policy (with which nodes decide whether or not to forward a packet to an awake node) needs to be carefully designed.

Exiting solutions that exploit path diversity attempt to address this issue by dealing with some local metrics. The anycast protocols in [10]–[12] let each node use the geographical distance from each neighboring node to the sink node to prioritize the forwarding decision to its neighboring nodes. The work in [13], [14] proposes anycast packet-forwarding protocols that work on top of a separate routing protocol in the network layer. The anycast protocols in [15]–[18] use the hop-count information (i.e., the number of hops for each node to reach the sink) such that at each hop the forwarding decision is chosen to reduce the hop count to the sink as soon as possible. However, these aforementioned approaches are heuristic in nature and do not directly minimize the expected end-to-end delay.

In our prior work [19], [20], we developed a distributed

This work has been partially supported by the National Science Foundation through awards CNS-0626703, CNS-0721477, CNS-0721434, CCF-0635202, and ARO MURI Award Nos. W911NF-07-10376 (SA08-03) and W911NF-08-1-0238. An earlier version of this paper has appeared in the Proceedings of IEEE INFOCOM, 2009 [1].

J. Kim and X. Lin are with School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN 47907, USA (Email:{jhkim,linx}@purdue.edu)

N. B. Shroff is with Department of Electrical and Computer Engineering and Department of Computer Science and Engineering, The Ohio State University, Columbus, OH 43210 (Email: shroff@ece.osu.edu).

P. Sinha is with Department of Computer Science and Engineering, The Ohio State University, Columbus, OH 43210 (Email: prasun@cse.ohio-state.edu).

anycast forwarding policy that simultaneously minimizes the expected end-to-end delays from all nodes to the sink, when the wake-up rates of the nodes are given. (The wake-up rate represents the frequency with which a node wakes up.) However, the delay-optimal anycast policy in [19], [20] was derived based on the assumption that nodes wake up according to a Poisson process (i.e., the wake-up intervals of a node are i.i.d. exponential random variables). Hence, the following questions remain unanswered: (1) If we can control the wake-up patterns (subject to given wake-up rates) in addition to the anycast forwarding policy, is there a wake-up pattern that results in optimal delay performance? and (2) If such a pattern exists, which forwarding policy is delay-optimal for the wake-up pattern? These questions make the problem more complex than the one considered [19], [20] because we can no longer exploit the memoryless property of a Poisson Process.

In this paper, we extend the results in [19], [20] to address these questions. For given wake-up rates of nodes (in other words, given energy budget at each node), we obtain the anycast forwarding policy and the wake-up pattern that can minimize the expected end-to-end delays from all nodes to the sink. Specifically, we show that using asynchronous periodic wake-up patterns along with an optimal forwarding policy can minimize the expected end-to-end delay over all asynchronous wake-up patterns. Further, we provide an efficient distributed algorithm that can implement the delay-optimal anycast forwarding policy for the periodic wake-up pattern.

The rest of this paper is organized as follows. In Section II, we describe our system model and formulate the delay-minimization problem that we intend to solve. In Section III, we study the delay-optimal anycast forwarding policy when nodes wake up periodically. In Section IV, we show that given an average wake-up rate, the periodic wake-up pattern is the best in terms of delay performance. In Section V, we provide simulation results that illustrate the superior performance of our proposed solution.

II. System Model

We consider an event-driven WSN with N sensor nodes. Let $\mathcal{N}$ be the set of all nodes. We assume in this paper that event information is reported to a single sink node s, but the analysis can be readily extended to the scenario with multiple sink nodes. Each node i has a set $\mathcal{N}_i$ of neighboring nodes to which node i is able to directly transmit packets.

The lifetime of an event-driven WSN under asynchronous sleep-wake scheduling consists of two phases: *the configuration phase* and *the operation phase*. When sensor nodes are deployed, the configuration phase begins, during which the nodes determine their packet-forwarding and sleep-wake scheduling policies. It is also during this phase that the optimization on these policies (which we will study in this paper) is carried out. Once the optimal policies are determined, the operation phase begins, during which the nodes apply the policies determined in the configuration phase to perform their main functions: detecting events and reporting the event information. Specifically, during this phase, sensor nodes alternate between sleeping and waking up, independently of other nodes. Consider a node that wakes up and hears a request from a neighboring node for relaying the event-reporting packets. If it is an eligible next-hop node based on the packet-forwarding policy, it receives the packet and then finds a new next-hop node to forward the packet. If the node successfully forwards the packets, it returns to sleep and follows the sleep-wake scheduling policy again.

A. Basic Forwarding and Sleep-Wake Scheduling Protocols

We first introduce the basic packet-forwarding and sleep-wake scheduling protocols that are used in the operation phase.

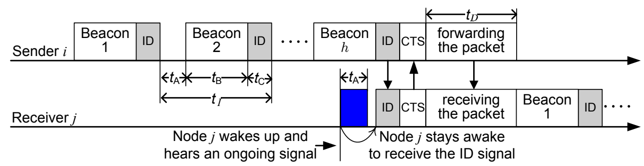


Fig. 1. System model

Packet-Forwarding Protocol: When a node i has a packet to deliver to the sink, it must wait for its neighboring nodes to wake up. Under asynchronous sleep-wake scheduling, we simply assume that the clocks at different nodes are not synchronized. Hence, the sending node i does not know exactly when its neighboring nodes will wake up (although it may have some statistical information of their wake-up patterns and wake-up rates).[1] Fig. 1 describes the protocol with which sending node i transmits its packet to one of its neighboring nodes. As soon as node i is ready to transmit the packet, it sends a beacon signal (Beacon 1 in Fig. 1) of duration t_B, and ID signal of duration t_C, and then listens for acknowledgements (CTS: Clear-To-Send) for duration t_A. The sending node repeats this sequence until it hears an acknowledgement. The ID signal contains the identity of the sending node and the sequence number of the last beacon signal. When a node wakes up and senses the h-th beacon signal, it will stay awake to decode the following ID signal, in which case we say that the node receives the h-th ID signal. (If a node wakes up in the middle of the ID signal, it must stay awake to decode the next ID signal.) Then, such a node has two choices. **Choice 1:** If the node chooses to receive the packet, it responds with a CTS message containing its identity during the acknowledgement period t_A that immediately follows the ID signal. Once the sending node hears the CTS, it forwards the packet to the awake node during the data transmission period t_D. **Choice 2:** If the awake node decides not to receive the packet, it goes back to sleep. For simplicity of notation, let $t_I = t_B + t_C + t_A$, which denotes the duration of each beacon-ID signaling iteration (See Fig. 1).

Remark: In the above basic protocol, we have ignored the possibility of collisions, which can be due to either multiple

[1] It may be possible for neighboring nodes to synchronize their clocks when they are forwarding event-reporting packets. However, since we assume that events occur rarely compared to the wake-up rates, by the time that the next event occurs, their clocks will drift substantially.

awake nodes or multiple sending nodes. In our on-line technical report [21, Section V], we describe an extended packet-forwarding protocol that addresses these collision scenarios using random or deterministic back-offs. However, in a low duty-cycle WSN where nodes seldom wake up, chances are small that multiple neighboring nodes wake up at the same beacon signal. Due to this reason, we use the basic protocol for our analysis and study the effect of collisions using simulation in Section V.

Sleep-Wake Scheduling Protocol: In order to save energy, each node wakes up infrequently and goes back to sleep if there is no activity in the neighborhood. Note that if the duration for which the node stays awake is shorter than t_A, the node may stay awake only within an acknowledgement period t_A and miss on-going beacon-ID signals. In order to avoid such a case, we assume that nodes must stay awake for at least t_A. Further, since a longer awake duration results in higher energy consumption, we set the awake duration to be exactly equal to t_A. The next time to wake up is determined by the sleep-wake scheduling policy of the node.

B. Sleep-Wake Scheduling and Anycast Forwarding Policies

In this subsection, we define the sleep-wake scheduling and anycast forwarding policies that are computed during the configuration phase and applied during the operation phase. These policies affect the end-to-end delay experienced by a packet, and the energy consumption of the network.

Sleep-Wake Scheduling Policy: Recall that, under asynchronous sleep-wake scheduling, nodes wake up independently of other nodes. Thus, the wake-up schedule of a node i can be seen as an independent random point process from the viewpoint of other nodes. We call this process the *wake-up process of node* i. Let $\#_i(t)$ be the number of times that node i has woken up in the time interval $[0,t]$. If a node j wakes up at time t and observe its neighboring node i for Δt amount of time, the number of times that node i wakes up within this period is given by $\#_i(t+\Delta t) - \#_i(t)$. However, since nodes do not synchronize their clocks with their neighboring nodes, the time t does not provide any further information on the distribution of $\#_i(t+\Delta t) - \#_i(t)$. Hence, we can assume that the wake-up process of a node i is as a stationary process from the viewpoint of other nodes, i.e., the distribution of $\#_i(t+\Delta t) - \#_i(t)$ does not depend on time t. We further assume that the wake-up process is ergodic, i.e., statistical properties of the wake-up process can be deduced from a sample path of the process (to be discussed in Section IV).

Wake-up Rate: We define the wake-up rate r_i of node i as the expected number of times that node i wakes up per unit time. Since the wake-up process is ergodic, the wake-up rate r_i must satisfy

$$\lim_{t\to\infty} \frac{\#_i(t)}{t} = r_i \text{ almost surely.} \tag{1}$$

Let $\vec{r} = (r_1, r_2, \cdots, r_N)$ be the global wake-up rate (or simply the wake rate). Note that a higher wake-up rate consumes energy faster.

Wake-up Pattern: For any stationary and ergodic wake-up process of a given node i, by re-scaling time, we can convert it to a process with a wake-up rate of 1. We define the wake-up pattern w_i as the control variable that fully characterizes this scaled wake-up process of node i. For example, if a node chooses *a periodic wake-up pattern* $w_i = w_{\text{per}}$, and its wake-up rate is r_i, it will wake up every $1/r_i$ time, i.e., the wake-up intervals are given by $1/r_i$. If *the Poisson wake-up pattern is chosen*, the intervals will be *i.i.d* exponential random variables with mean $1/r_i$. Node i can also choose a wake-up pattern such that the wake-up intervals are correlated, e.g., node i can alternate with the wake-up intervals of length $\frac{1}{2r_i}$ and $\frac{3}{2r_i}$. Let $\vec{w} = (w_1, w_2, \cdots, w_N)$ denote the global wake-up pattern (or simply the wake-up pattern).

Remark: While the wake-up rate determines the expected wake-up interval, the wake-up pattern determines the distribution of the interval. Hence, the wake-up rate and the wake-up pattern of a node fully determine the wake-up process (schedule) of the node.

Anycast Forwarding Policy: Suppose that a sending node i has sent the h-th beacon-ID signal, and a set $X \subset \mathcal{N}_i$ of the neighboring nodes wakes up and receives the ID signal. We let $f_{i,h}(X)$ denote the corresponding decision of the sending node i, which is to be specified next. We let $f_{i,h}(X) = j$ if the sending node i decides to transmit the packet to node $j \in X$, and we let $f_{i,h}(X) = i$ if the sending node i decides to send out the $(h+1)$-st beacon-ID signal, i.e., the packet remains at node i. This forwarding decision may seem inconsistent with the packet-forwarding protocol decribed in Subsection II-A, in which the sending node is restricted to transmit the packet whenever it receives a CTS. However, we only use this more general setting to find the optimal forwarding decisions and then show that such optimal decisions can be implemented by our packet-forwarding protocol that lets the sending node always transmit the packet whenever it receives a CTS. Let $f_i = \{f_{i,1}, f_{i,2}, \cdots\}$ denote the anycast forwarding policy of node i (or simply the anycast policy of node i). We further denote by $f = \{f_1, f_2, \cdots, f_N\}$ the global anycast forwarding policy (or simply the anycast policy).

Remark: In [19], [20], the wake-up pattern is assumed to be Poisson. Due to the memoryless property of the Poisson wake-up pattern, the probability that each neighboring node wakes up at a beacon-ID signal does not change with the number h of beacon-ID signals sent. Hence, the optimal forwarding decisions must also be the same at each iteration, i.e., $f_{i,1} = f_{i,2} = \cdots$. In contrast, since we remove the Poisson assumption in this paper, we have to also consider policies that change with h, i.e., $f_{i,1} \neq f_{i,2} \neq \cdots$.

C. Performance Metrics and Optimization

In this section, we define the notion of the end-to-end delay. We then formulate the problem of minimizing the end-to-end delay by jointly controlling the anycast forwarding policy and the sleep-wake scheduling policy.[2]

Expected end-to-end delay: During the operation phase, we define the end-to-end delay as the delay from the time

[2] As mentioned earlier, our goal during the configuration phase is to design the system to minimize the delay of interest during the operation phase.

when a source node detects an event and generates the event-reporting packet (or packets) to the time the *first* packet is received at the sink. For applications that use a single packet to carry the event information, the above definition captures the actual delay for reporting the event information. For applications that use multiple packets, if the nodes that relayed the first packet stay awake for a while, the delay to relay subsequent packets will be much smaller than that experienced by the first packet. (For instance, these subsequent packets may be sent a few nodes behind the first packet, and hence they can reach the sink soon after the first packet reaches the sink.) Hence, the actual event-reporting delay can still be approximated by the delay experienced by the first packet.

The sleep-wake scheduling policy $(\vec{r},\vec{w})$ and anycast forwarding policy f fully determine the stochastic process with which the first packet traverses the network from the source node to the sink. Hence, we use $D_i(\vec{r},\vec{w},f)$ to denote the expected end-to-end delay from node i to the sink under the joint policy $(\vec{r},\vec{w},f)$. For simplicity, from now on, we simply call the expected end-to-end delay from node i to the sink as "the delay from node i."

Delay-Minimization Problem: The objective of this paper is to find the optimal joint policy $(\vec{w},f)$ that solves the following delay-minimization problem for given wake-up rate $\vec{r}$:

$$\min_{\vec{w},f} D_i(\vec{r},\vec{w},f). \tag{2}$$

Note that $\vec{r}$ controls the duty cycle of the sensor network, which in turn controls the energy expenditure. Hence, the problem can also be viewed as minimizing the delays for a given energy budget. In Sections III and IV, we develop an algorithm that solves this problem for all nodes i, i.e., our solution can simultaneously minimize the delays from all nodes.

III. Delay-Optimal Anycast Policy for a given Sleep-Wake Scheduling Policy

As a preliminary step to solving the delay-minimization problem, in this section we first fix a sleep-wake scheduling policy $(\vec{r},\vec{w})$ and study delay-optimal anycast policies for the fixed sleep-wake scheduleing policy. This problem can be formulated as a stochastic shortest path (SSP) problem, where the state corresponds to the node that is holding the packet, and the cost corresponds to the delay for each packet to reach the sink. In Section III-A, we will derive a solution to this problem, by using the value-iteration algorithm. A key part of the value-iteration algorithm is, assuming that node i knows the end-to-end delay from its neighboring nodes to the destination, how node i should update its own forwarding policy. This corresponds to a sub-problem, in which the sending node needs to decide whether to forward the packet to an awake node, or to send the next beacon signal and wait for another node to wake up. This problem can again be formulated as an infinite-horizon dynamic programming problem where the state corresponds the set of awake nodes after each beacon signal. We derive the solution to this sub-problem in Section III-B. However, the optimal policy in Sections III-A and III-B can be difficult to compute in practice due to the infinite horizon. In Section III-C, we proposed a more practical truncated version of the forwarding policy, and show that the optimal truncated policy will converge to the original optimal policy as a parameter approaches infinity. Finally, in Section III-D, we study the important properties of periodic wakeup patterns and show that the truncated policy becomes exactly optimal under the periodic wake-up pattern.

A. Value-Iteration Algorithm

In this subsection, we develop the value-iteration algorithm. Given a sleep-wake scheduling policy $(\vec{r},\vec{w})$, the delay-minimization problem can be formulated as a stochastic shortest path (SSP) problem [22, Chapter 2], where the sensor node that has a packet corresponds to the "state", and the delay corresponds to the "cost" that we intend to minimize. The sink s corresponds to the terminal state, where no further cost (delay) will be incurred. Let $i_0, i_1, i_2, \cdots, i_L = s$ be the sequence of nodes that relay the packet from the source node i_0 to the sink s in L steps. Note that under anycasting, this sequence is random because each node has a set of candidate next-hop nodes and does not know which of them will wake up first to receive the packet. Let $D_{\text{hop},i}(\vec{r}_i,\vec{w}_i,f_i)$ be the expected one-hop delay at node i under the forwarding policy f_i, where the wake-up rates and patterns of neighboring nodes are given by $\vec{r}_i \triangleq (r_j, j\in\mathcal{N}_i)$ and $\vec{w}_i \triangleq (w_j, j\in\mathcal{N}_i)$. We note that the wake-up rates and patterns of the other nodes not in $\mathcal{N}_i$ do not affect the one-hop delay of node i. Then, the end-to-end delay $D_i(\vec{r},\vec{w},f)$ from each node i_0 to the sink can be expressed as

$$D_i(\vec{r},\vec{w},f) = E\Big\{\sum_{l=0}^{L} D_{\text{hop},i_l}(\vec{r}_{i_l},\vec{w}_{i_l},f_{i_l})\Big\}, \tag{3}$$

where the expectation is taken with respect to the random sequence $i_1, i_2, \cdots, i_L$. Given the sleep-wake scheduling policy $(\vec{r},\vec{w})$, let $D_i^*(\vec{r},\vec{w}) \triangleq \min_f D_i(\vec{r},\vec{w},f)$ be the minimum expected delay from node i. Then, according to the Bellman equation [22, Section 2.2], for all nodes i, the minimum delay $D_i^*(\vec{r},\vec{w})$ of node i must satisfy

$$D_i^*(\vec{r},\vec{w}) = \min_{f_i}\Big(D_{\text{hop},i}(\vec{r}_i,\vec{w}_i,f_i) + \sum_{j\in\mathcal{N}_i} q_{i,j}(\vec{r}_i,\vec{w}_i,f_i) D_j^*(\vec{r},\vec{w})\Big), \tag{4}$$

where $q_{i,j}(\vec{r}_i,\vec{w}_i,f_i)$ is the probability that node j is chosen as the next-hop node of node i under the forwarding policy f_i. Further, using the following value-iteration algorithm [22, Section 1.3], we can find the delay-optimal forwarding policy that achieves $D_i^*(\vec{r},\vec{w})$ for all nodes i:

Value Iteration Algorithm: At the initial iteration $k = 0$, all nodes i set their initial delay values $D_i^{(0)}$ to ∞, and the sink s sets its delay value $D_s^{(0)}$ to zero. At each iteration $k = 1, 2, \cdots$, every node i collects the delay values $D_j^{(k-1)}$ from its neighboring nodes j and then updates its delay value $D_i^{(k)}$ by solving

$$D_i^{(k)} = \min_{f_i}\Big(D_{\text{hop},i}(\vec{r}_i,\vec{w}_i,f_i) + \sum_{j\in\mathcal{N}_i} q_{i,j}(\vec{r}_i,\vec{w}_i,f_i) D_j^{(k-1)}\Big). \tag{5}$$

TABLE I
TABLE OF NOTATIONS

Notation	Definition
D_j	Expected delay from neighboring node j to the sink
X_h	Set of awake neighboring nodes right after beacon signal h
x_h	Node j that has the smallet delay value D_j among the nodes in X_h
$P^{(h)}_{x,x'}$	Conditional probability that $x_h = x'$ conditioned on that $x_{h-1} = x$ and node i sends beacon signal h
$p_{j,h}$	Conditional probability that node j wakes up at stage h conditioned on not having woken up at earlier stages
$d^{(h)}(x_h)$	Expected delay after node i sends beacon signal h conditioned on that the current state is x_h, and the optimal forwarding policy will be used afterward
$d^{(h)}_{\text{wait}}(x_h)$	Expected delay after node i sends beacon signal h conditioned on that the current state is x_h, and node i sends beacon signal $h+1$ and uses the optimal forwarding policy afterward

Let $f_i^{(k)}$ be the forwarding policy of node i that minimizes (5). Then, according to [22, Proposition 2.2.2], the delay value $D_i^{(k)}$ of each node i converges to the minimum delay $D_i^*(\vec{r},\vec{w})$, i.e., $\lim_{k\to\infty} D_i^{(k)} = D_i^*(\vec{r},\vec{w})$, and the corresponding forwarding policy $f^{(k)} = \{f_1^{(k)}, f_2^{(k)}, \cdots\}$ also converges to the delay-optimal forwarding policy, i.e., $\lim_{k\to\infty} f^{(k)} \in \arg\min_f D_i(\vec{r},\vec{w},f)$ for all nodes i.

The key step in this value iteration algorithm is how every node i solves the sub-problem in (5) at each iteration k. Note that this subproblem is equivalent to the following problem: we need to find a forwarding policy of node i that minimizes the expected delay from node i when the delays from neighboring nodes j to the sink are given by $D_j^{(k-1)}$, and the sleep-wake scheduling policies of neighboring nodes are given by $(\vec{r}_i, \vec{w}_i)$. In the next two subsections, we will develop the LOCAL-OPT algorithm that solves this sub-problem.

B. LOCAL-OPT Algorithm

To solve the above sub-problem, we focus on a node i that has a packet. For ease of exposition, let the expected delays from neighboring nodes j be denoted by $D_j = D_j^{(k-1)}$ ($j \in \mathcal{N}_i$), which is equal to $D_j^{(k-1)}$ for iteration k in the value-iteration algorithm. Without loss of generality, we assume that the node i has neighboring nodes $1, 2, \cdots, N_i$ ($N_i = |\mathcal{N}_i|$), and their expected delays are sorted in increasing order, i.e., $D_1 \le D_2 \le \cdots \le D_{N_i}$. To avoid confusion, we further assume that the index i of the sending node is larger than N_i+1. In Table I, we summarize the definition of the notations that will be used in this section.

After the sending node i sends out the h-th beacon signal, it has to choose either to **transmit** the packet to one of the awake nodes, or to **wait** for the other node to wake up by sending the next beacon signal. We call this moment the decision stage h (or simply stage h) and denote the set of the awake nodes at this moment by X_h. By definition, $f_{i,h}(X_h) = j$ ($j \in X_h$) implies that node i decides to transmit to node j, and $f_{i,h}(X_h) = i$ implies that node i decides to wait and send the $(h+1)$-st beacon signal. Since stage 0 is the moment when node i is about to send the first beacon signal, we set $X_0 = \emptyset$ and $f_{i,0}(X_0) = i$. This state transition terminates whenever the sending node transmits the packet to an awake node j. When this happens, the packet will be relayed by the node j and eventually arrive at the sink after D_j time (the delay from node j). We denote this terminal state by state 0.

Since the number of possible states at each stage increases exponentially with the number N_i of neighboring nodes (2^{N_i} states at each stage), it is more convenient to deal with a simpler transition model as follows. Note that if node i decides to transmit the packet to one of the awake nodes in X_h, clearly it should choose the node j with the smallest delay D_j among all the awake nodes in order to minimize the delay from the next-hop node to the sink. Hence, at each stage h, node i only needs to remember the awake node with the smallest delay. In other words, if a delay-optimal policy is applied, only the awake node with the smallest delay affects the state transition dynamics. We denote this node by $x_h = \arg\min_{j\in X_h} D_j$. (Ties are broken arbitrarily.) If no nodes are awake ($X_h = \emptyset$), we denote the corresponding state x_h by $x_h = N_i + 1$. (For example, since $X_0 = \emptyset$, the initial state is always given by $x_0 = N_i + 1$.) From now on, we can use a simpler state transition model $x_0, x_1, x_2, \cdots$ to solve the sub-problem (5) without any loss of optimality. Due to the same principle, we abuse notation slightly, and use $f_{i,h}(x_h)$ to denote the decision of node i at state x_h as follows: $f_{i,h}(x_h) = x_h$ if the sending node i decides to transmit the packet to node x_h, and $f_{i,h}(x_h) = i$ if the node i decides to wait. We further use the following assumption to simplify the dynamics for the state transitions $x_0, x_1, \cdots$. (However, this is not a required assumption, as we will soon see.)

Assumption 1: If an awake node is not chosen as the next-hop node, we assume that the node stays awake to remain eligible to be chosen as the next-hop node at following stages. Under this assumption, the state transition must satisfy $x_0 \ge x_1 \ge \cdots$.

Remark: Assumption 1 not only simplifies the analysis, but it also clearly leads to smaller delay, compared to the case where an awake node can return to sleep when it is not immediately chosen as the next-hop node. However, one could argue that keeping nodes awake consumes more energy. In Section III-D, we will show that the optimal anycast forwarding policy achieves the minimum delay without Assumption 1, and thus the awake nodes in fact do not need to stay awake. But for now, we use the assumption to simplify the analysis.

We next consider the state transition probability. Let $P^{(h)}_{x,x'}$ be the state transition probability from state $x_{h-1} = x$ to state $x_h = x'$, given that node i decides to wait at stage $h-1$, i.e., $P^{(h)}_{x,x'} \triangleq \Pr\big(x_h = x' | x_{h-1} = x \text{ and } f_{i,h-1}(x) = i\big)$. Let $p_{j,h}$ be the conditional probability that node j wakes up at stage h conditioned on not having woken up at earlier stages. Using $p_{j,h}$, we can express the state transition probability as

$$P^{(h)}_{x,x'} = \begin{cases} p_{x',h}\prod_{j=1}^{x'-1}(1-p_{j,h}) & \text{if } x' < x, \\ \prod_{j=1}^{x'-1}(1-p_{j,h}) & \text{if } x' = x, \\ 0 & \text{otherwise.} \end{cases} \tag{6}$$

The state transition probability conditioned on $f_{i,h-1}(x) = x$ is trivial because, if the sending node decides to transmit the packet to node x, the next state must be 0. Note that if the wake-up pattern of node 1 is such that node 1 must wake up before beacon signal h, the probability $P_{x,x'}^{(h)}$ is not well defined for $x > 1$ because the conditional event $x_{h-1} = x$ cannot happen. Hence, we say that state $x_h = x$ is *admissible* if $\Pr(x_h = x | f_{i,h'}(x_{h'}) = i, \forall h' < h) > 0$, and we define the state transition probability only for admissible states. We also define $x_{h,\max}$ as the upperbound of the admissible state at stage h, i.e., $x_h \le x_{h,\max}$.

In our dynamic programming problem, the cost to be minimized is delay. Let $g(x_h, f_{i,h}(x_h))$ be the one-step delay between stages h and $h+1$ when decision $f_{i,h}$ is used at state x_h. If the sending node i sends out the next beacon signal ($f_{i,h}(x_h) = i$), the delay incurred by this decision is the beacon-ID duration t_I. If node i transmits the packet, the packet will be transmitted to the next-hop node x_h for the packet transmission period t_D and will arrive at the sink D_{x_h} time later. Hence, the delay incurred by this decision is $t_D + D_{x_h}$. Once the packet reaches the sink, no more delay will be incurred. Hence, the one-step delay can be expressed as

$$g(x_h, f_{i,h}(x_h)) = \begin{cases} t_I & \text{if } f_{i,h}(x_h) = i, \quad (7) \\ t_D + D_{x_h} & \text{if } f_{i,h}(x_h) = x_h. \quad (8) \end{cases}$$

for $x_h \neq 0$ and $g(x_h, f_{i,h}(x_h)) = 0$ for $x_h = 0$. Using the above state transition probability and the one-step delay, we can represent the sub-problem (5) as the following infinite-horizon dynamic program (DP) problem [22, Chapter 1]: given the delays D_j from the neighboring nodes j, we want to find the anycast forwarding policy f_i of node i that minimizes the overall cost (delay) function

$$d_{f_i} = \lim_{\bar{h}\to\infty} E\left\{\sum_{h'=0}^{\bar{h}-1} g(x_{h'}, f_{i,h'}(x_{h'}))\right\} \tag{9}$$

where $x_0, x_1, x_2, \cdots$ are the states visited, and the expectation is taken with respect to these states. Then, $\min_{f_i} d_{f_i}$ and $\arg\min_{f_i} d_{f_i}$ corresponds to $D_i^{(k)}$ and $f_i^{(k)}$ of the value-iteration algorithm in (5), respectively.

To solve this DP problem, we define $d^{(h)}(x_h)$ as the expected delay from state $x_h \ge 1$ at stage h, given that the optimal forwarding policy is applied afterward, i.e.,

$$d^{(h)}(x_h) \triangleq \min_{f_{i,h}, f_{i,h+1}, \cdots} \left(\lim_{\bar{h}\to\infty} E\left\{\sum_{h'=h}^{\bar{h}-1} g(x_{h'}, f_{i,h'}(x_{h'}))\right\}\right)$$

where $x_{h+1}, x_{h+2}, \cdots$ are the states to be visited after stage h, and the expectation is taken with respect to these states. By definition, it immediately follows that $d^{(0)}(N_i + 1) = \min_{f_i} d_{f_i}$. The delay function $d^{(h)}(x_h)$ can be interpreted as the minimum expected delay from state x_h. Suppose that the sending node i at state x_h decides to **transmit** the packet to node x_h ($f_{i,h}(x_h) = x_h$). By (7), the minimum expected delay conditioned on this decision is $t_D + D_{x_h}$. If node i decides to **wait** ($f_{i,h}(x_h) = i$), the minimum expected delay $d_{\text{wait}}^{(h)}(x_h)$ conditioned on this decision is given by

$$d_{\text{wait}}^{(h)}(x_h) = t_I + \sum_{x_{h+1}=1}^{x_h} P_{x_h, x_{h+1}}^{(h+1)} d^{(h+1)}(x_{h+1}), \tag{10}$$

assuming that the optimal decisions $f_{i,h+1}^*, f_{i,h+2}^*, \cdots$ are applied afterward. Then, the optimal forwarding decision at stage h is the one that incurs a smaller delay [22, Equation (1.3) on Page 5], i.e.,

$$f_{i,h}^*(x_h) = \begin{cases} x_h & \text{if } t_D + D_{x_h} < d_{\text{wait}}^{(h)}(x_h), \\ i & \text{otherwise.} \end{cases} \tag{11}$$

Further, the minimum expected delay $d^{(h)}(x_h)$ at stage h is given by

$$d^{(h)}(x_h) = \min(d_{\text{wait}}^{(h)}(x_h), t_D + D_{x_h}). \tag{12}$$

Although (11) and (12) are not well defined for $x_h = N_i + 1$, by setting $D_{N_i+1} = \infty$, we can still use (11) and (12) even when $x_h = N_i + 1$. In this case, $d^{(h)}(N_i + 1)$ is always equal to $d_{\text{wait}}^{(h)}(N_i + 1)$. (In other words, if no nodes are awake, the only choice left is to send the next beacon-ID signal.)

Clearly, whenever node 1 has woken up, the optimal decision is to forward the packet to node 1. Hence, the optimal forwarding decision must satisfy

$$f_{i,h}^*(1) = 1 \text{ and } d^{(h)}(1) = t_D + D_1 \text{ for all } h. \tag{13}$$

Furthermore, since the packet will be forwarded to a neighboring node j eventually, taking $t_D + D_j$ expected time, it must hold that

$$d^{(h)}(x_h) \ge t_D + D_1 \text{ for } x_h > 0. \tag{14}$$

We have shown that for an arbitrary sleep-wake process the optimal forwarding decision $f_{i,h}^*$ and the delay value $d^{(h)}$ must satisfy the necessary conditions in (11) and (12), respectively. If there is a reference stage $\bar{h}$ such that the minimum delay $d^{(\bar{h})}(x_{\bar{h}})$ is known for all admissible states $x_{\bar{h}}$, we can then use (10) and (12) as a backward iteration from stage $\bar{h}$ to stage 0, and can find the optimal forwarding decisions. However, such a reference stage may not exist in general. In practice, we can artificially impose a reference stage $\bar{h}$ and use a truncated policy after $\bar{h}$. In the next section, we will study the performance of such a truncated packet-forwarding policy as $\bar{h} \to \infty$.

C. A Truncated Forwarding Policy

We use $\hat{f}_{i,h}$ to denote a packet-forwarding policy that uses truncated decisions after a given stage $\bar{h}$. In the rest of the paper, we refer to it as *the truncated policy*. Specifically, if the sending node has not chosen its next-hop node until stage $\bar{h}$, it then waits only for node 1 (the node with the smallest delay) to wake up and then forwards the packet to node 1. Let H be the number of beacon signals that the sending node has to send until node 1 wakes up. Then, if node 1 has not woken up for the first $\bar{h}$ beacon signals, i.e, $x_{\bar{h}} > 1$, the sending node has to send $H - \bar{h}$ more beacon signals until node 1 wakes up. Similar to $d^{(h)}(x_h)$, we define $\hat{d}^{(h)}(x_h)$ as the expected delay from state x_h at stage h under the truncated policy. Then, the expected delay $\hat{d}^{(\bar{h})}(x_{\bar{h}})$ at stage $\bar{h}$ is given by

$$\hat{d}^{(\bar{h})}(x_{\bar{h}}) = \begin{cases} t_D + D_1 & \text{if } x_{\bar{h}} = 1, \\ E[H - \bar{h} | H > \bar{h}] \cdot t_I + t_D + D_1 & \text{if } x_{\bar{h}} > 1. \end{cases} \tag{15}$$

Since we now know the value of $\hat{d}^{(\bar{h})}(x_{\bar{h}})$ for all admissible states $x_{\bar{h}}$ at stage $\bar{h}$, we can compute the optimal forwarding decision at stages $h < \bar{h}$ for the truncated policy. Similarly to (10) and (12), we compute $\hat{d}^{(h)}_{\text{wait}}(x_h)$ (the minimum expected delay conditioned on the WAIT decision) and $\hat{d}^{(h)}(x_h)$ for $h = \bar{h}-1, \bar{h}-2, \cdots, 1, 0$, using

$$\hat{d}^{(h)}_{\text{wait}}(x_h) = t_I + \sum_{x_{h+1}=1}^{x_h} P^{(h+1)}_{x_h,x_{h+1}} \hat{d}^{(h+1)}(x_{h+1}), \tag{16}$$

and

$$\hat{d}^{(h)}(x_h) = \min(\hat{d}^{(h)}_{\text{wait}}(x_h), t_D + D_{x_h}). \tag{17}$$

Once we obtain these values, the optimal truncated policy can be expressed as follows:

$$\hat{f}_{i,h}(x_h) = \begin{cases} 1 & \text{if } x_h = 1, \\ x_h & \text{if } x_h > 1, h < \bar{h}, \text{and} \\ & D_{x_h} < \hat{d}^{(h)}_{\text{wait}}(x_h) - t_D, \\ i & \text{otherwise.} \end{cases} \tag{18}$$

Since the delay under the truncated policy cannot be smaller than that under the optimal policy, we have

$$d^{(h)}(x_h) \leq \hat{d}^{(h)}(x_h), \tag{19}$$

for all h and admissible states x_h. Note that $\hat{d}^{(0)}(N_i+1)$ corresponds to the expected delay of the sending node under the truncated policy, and $d^{(0)}(N_i+1)$ corresponds to that under the optimal forwarding policy. In the following proposition, we show that the delay gap between the optimal and the truncated forwarding policies will approach to zero as $\bar{h} \to \infty$.

Proposition 1: The truncated forwarding policy $\hat{f}_i$ has the following properties:
(a) $\hat{d}^{(0)}(N_i+1) - d^{(0)}(N_i+1) \leq \Pr(H > \bar{h}) E[H - \bar{h} | H > \bar{h}] \cdot t_I$,
(b) $\hat{d}^{(0)}(N_i+1) - d^{(0)}(N_i+1) \to 0$ as $\bar{h} \to \infty$.

Proof: We first show by induction that

$$\hat{d}^{(h)}(x_h) - d^{(h)}(x_h) \leq \Pr(H > \bar{h}|x_h) E[H - \bar{h}|H > \bar{h}] \cdot t_I. \tag{20}$$

holds for $h \leq \bar{h}$ and all admissible states $x_h > 0$. At stage $\bar{h}$, if $x_{\bar{h}} = 1$, we have $\hat{d}^{(\bar{h})}(1) - d^{(\bar{h})}(1) = 0$ from (13) and (15), and thus (20) holds. If $x_{\bar{h}} > 1$, from (14), it holds that $d^{(\bar{h})}(x_{\bar{h}}) \geq t_I + D_1$. Hence, using (15) and (19), we have

$$\hat{d}^{(\bar{h})}(x_{\bar{h}}) - d^{(\bar{h})}(x_{\bar{h}}) \leq E[H - \bar{h}|H > \bar{h}] \cdot t_I.$$

Since $x_{\bar{h}} > 1$, i.e., node 1 has not woken up until stage $\bar{h}$, we have $\Pr(H > \bar{h}|x_{\bar{h}}) = 1$. Hence, (20) holds for $h = \bar{h}$.

We now assume that (20) holds for stage $h+1$. Since $\hat{d}^{(h+1)}_{\text{wait}}(1) = d^{(h+1)}_{\text{wait}}(1)$, using (10) and (16), we have

$$\begin{aligned} &\hat{d}^{(h)}_{\text{wait}}(x_h) - d^{(h)}_{\text{wait}}(x_h) \\ &\leq \sum_{x_{h+1}=2}^{x_h} P^{(h+1)}_{x_h,x_{h+1}} \Pr(H > \bar{h}|x_{h+1}) E[H - \bar{h}|H > \bar{h}] \cdot t_I \\ &= \Pr(H > \bar{h}|x_h) E[H - \bar{h}|H > \bar{h}] \cdot t_I. \end{aligned} \tag{21}$$

From (12) and (17), we have $\hat{d}^{(h)}(x_h) - d^{(h)}(x_h) \leq \hat{d}^{(h)}_{\text{wait}}(x_h) - d^{(h)}_{\text{wait}}(x_h)$. Hence, from (21), Inequality (20) holds for h. Then, by induction, (20) holds for all $h = 0, 1, \cdots, \bar{h}$.

Since $x_0 = N_i + 1$ with probability 1, it holds that $\Pr(H > \bar{h}|x_0 = N_i+1) = \Pr(H > \bar{h})$. Hence, for $h = 0$, we have

$$\begin{aligned} &\hat{d}^{(0)}(N_i+1) - d^{(0)}(N_i+1) \\ &\leq \Pr(H > \bar{h}) E[H - \bar{h}|H > \bar{h}] \cdot t_I \quad (22) \\ &= E[(H - \bar{h}) 1_{\{H > \bar{h}\}}] \cdot t_I, \quad (23) \end{aligned}$$

where $1_{\{\cdot\}}$ is an indicator function. From (22), Property (a) follows. Since $E[H] < \infty$, (23) must converge to 0 as $\bar{h}$ increases. Hence, Property (b) follows. ∎

Proposition 1 implies that (a) *the truncated forwarding policy is asymtotically optimal*, and (b) the rate of convergence depends on the decay rate of the tail probability $\Pr(H > \bar{h})$. If nodes, for instance, wake up according to the Poisson wake up pattern, $E[H - \bar{h}|H > \bar{h}]$ will be given by a constant because of the memoryless property, and the probability $\Pr(H > \bar{h})$ will decay exponentially. Hence, the delay gap between the truncated policy and the optimal policy will decrease exponentially.

Although we can compute the optimal truncated policy $\hat{f}_i$, it is still difficult to implement such a policy because of the following reasons. First, the policy requires the sender to know the list (X_h or x_h) of awake nodes at each stage h. It can be difficult for the sender to acquire this information during a short period t_A between two beacon-ID signals because of collisions. Second, the optimal policy is based on Assumption 1, which requires that an awake node stay awake even if it is not immediately chosen as the next-hop node. However, if the node is not chosen as the next-hop node in the end, the additional energy that it has spent to remain awake is then wasted. The following proposition contains an important result to address the above implementation issues.

Proposition 2: For $h = 0, 1, \cdots, \bar{h}-1$ and all admissible states $x_h = x', x''$ such that $1 < x' \leq x''$, we have

$$\hat{d}^{(h)}_{\text{wait}}(x'') - \hat{d}^{(h)}_{\text{wait}}(x') \leq D_{x''} - D_{x'}. \tag{24}$$

Proof: We prove this result by induction. For $h = \bar{h}-1$, by applying (15) to (16), we can verify that $\hat{d}^{(h)}_{\text{wait}}(x'')$ is equal to $\hat{d}^{(h)}_{\text{wait}}(x')$ for all states $x'' \geq x' > 1$. Hence, (24) holds for $h = \bar{h}-1$.

We now assume that (24) holds for stage $h+1 \leq \bar{h}-1$. From (17), we also have

$$\hat{d}^{(h+1)}(x'') - \hat{d}^{(h+1)}(x') \leq D_{x''} - D_{x'}. \tag{25}$$

for $0 < x' \leq x''$. Using (6) and (10), we have

$$\begin{aligned} &\hat{d}^{(h)}_{\text{wait}}(x'') - \hat{d}^{(h)}_{\text{wait}}(x') \\ &= \sum_{x_{h+1}=x'}^{x''} P^{(h+1)}_{x'',x_{h+1}} d^{(h+1)}(x_{h+1}) - P^{(h+1)}_{x',x'} d^{(h+1)}(x'). \end{aligned}$$

Note that $\sum_{x_{h+1}=x'}^{x''} P^{(h+1)}_{x'',x_{h+1}} = P^{(h+1)}_{x',x'}$ from (6). Hence, the R.H.S. of the above can be expresses as

$$\begin{aligned} &\sum_{x_{h+1}=x'}^{x''} P^{(h+1)}_{x'',x_{h+1}} (d^{(h+1)}(x_{h+1}) - d^{(h+1)}(x')) \\ &\leq \sum_{x_{h+1}=x'}^{x''} P^{(h+1)}_{x'',x_{h+1}} (D_{x_{h+1}} - D_{x'}) \leq D_{x''} - D_{x'}, \end{aligned}$$

where in the last step we have used (25). Hence, (24) also holds for stage h. By induction, the result follows. ■

Using Proposition 2, the sending node i can implement the optimal truncated policy as follows during the operation phase: **Implementation of the optimal truncated policy:** In the configuration phase, every neighboring node i computes the set $h_j^{(i)}$ of beacon signals for $j \in \mathcal{N}_i$ such that

$$h_j^{(i)} \triangleq \{h < \bar{h} \mid j \leq x_{h,\max}, D_j < d_{\text{wait}}^{(h)}(j) - t_D\} \tag{26}$$

and then informs $h_j^{(i)}$ to each neighboring node j. In the operation phase, if node j wakes up and hears beacon signal h from node i, it sends a CTS if and only if $h \in h_j^{(i)}$. If $h \notin h_j^{(i)}$, node j returns to sleep and wakes up at the next beacon signal in $h_j^{(i)}$. Among the neighboring nodes that have sent a CTS, the sending node i forwards the packet to the node j with the smallest delay value D_j.

We now show that the above method implements the optimal truncated policy. If node j wakes up and hears beacon signal h, the current state x_h must be at least state j, i.e., $x_h \leq j$. We now consider three cases. **Case (A):** If $j > x_{h,\max}$, there must exist another awake node that has a smaller delay value than D_j. Hence, node j has no chance to be a next-hop node, and thus it does not need to respond. **Case (B-1):** If $j \leq x_{h,\max}$ and $D_j < \hat{d}_{\text{wait}}^{(h)}(j) - t_D$, it immediately follows from Proposition 2 that $D_{x_h} < \hat{d}_{\text{wait}}^{(h)}(x_h) - t_D$. Hence, from (18), the decision must be $\hat{f}_{i,h}(x_h) = x_h$, and node x_h will receive the packet. (If $x_h = 1$, in which case Proposition 2 does not apply, we still have $\hat{f}_{i,h}(x_h) = x_h$ from (18).) In the above implementation, since both x_h and j will respond, the correct decision is reached. **Case (B-2):** If $j \leq x_{h,\max}$ and $D_j \geq d_{\text{wait}}^{(h)}(j) - t_D$, node j cannot be the next-hop node according to the truncated policy in (18). Hence, node j does not need to respond, and it can wait until the next beacon signal h' such that $D_{h'} < d_{\text{wait}}^{(h')}(j) - t_D$. *From all the cases* ***(A)****,* ***(B-1)****, and* ***(B-2)****, we can conclude that the above method exactly implements the optimal truncated policy, and does not require for the sending node to know the current state x_h. However, we still need Assumption 1 because node j in case* ***(B-2)*** *has to wake up at a later beacon signal.* In the next subsection, we will show that when all neighboring nodes wake up periodically, Assumption 1 is not even necessary for the implementation.

We below summarize the value-iteration algorithm and the LOCAL-OPT algorithm that every node i runs during the configuration phase.

Value-Iteration Algorithm

$D_i^{(0)} \leftarrow \infty$
for $k = 1$ to $k_{\max}$ **do**
  Collect $D_j^{(k-1)}$ from neighboring nodes j
  $(D_i^{(k)}, (h_j^{(i)}, j \in \mathcal{N}_i)) \leftarrow$ **LOCAL-OPT**$((D_j^{(k-1)}, j \in \mathcal{N}_i))$
end for
return $D_i^{(k)}$,$(h_j^{(i)}, j \in \mathcal{N}_i)$

LOCAL-OPT Algorithm

Receive $(D_j^{(k-1)}, j \in \mathcal{N}_i)$
Sort $(D_j^{(k-1)}, j \in \mathcal{N}_i)$ in an increasing order
Let $D_1, D_2, \cdots, D_{N_i}$ be the sorted delay and $m(1), m(2), \cdots, m(N_i)$ be the corresponding node indices.
Set $\bar{h}$
for $j = 1$ to $N_i + 1$ **do**
  Set $\hat{d}^{(\bar{h})}(j)$ using (15)
  $h_j^{(i)} \leftarrow \emptyset$
end for
for $h = \bar{h} - 1$ to 0 **do**
  for $j = 1$ to $x_{h,\max}$ **do**
    Compute $\hat{d}_{\text{wait}}^{(h)}(j)$ using (16)
    if $D_j < d_{\text{wait}}^{(h)}(j) - t_D$ **then**
      $h_{m(j)}^{(i)} \leftarrow h_{m(j)}^{(i)} \cup \{h\}$
    end if
    $\hat{d}^{(h)}(j) \leftarrow \min(\hat{d}_{\text{wait}}^{(h)}(j), t_D + D_{x_h})$
  end for
end for
$h_{m(1)}^{(i)} \leftarrow h_{m(1)}^{(i)} \cup \{\bar{h}, \bar{h}+1, \cdots\}$
return $\hat{d}^{(0)}(N_i + 1)$, $(h_j^{(i)}, j \in \mathcal{N}_i)$

During the operation phase that follows the configuration phase, each node j uses the implementation for the optimal truncated policy.

Sleep-wake Scheduling Protocol

1: **loop**
2:   Set up the next time t_{wake} that node j has to wake up according the sleep-wake scheduling policy (r_j, w_j).
3:   Wake up at time t_{wake}.
4:   **if** Hear beacon signal h from a neighboring node i **then**
5:     **if** $h \in h_j^{(i)}$ **then**
6:       Respond a CTS signal to the sending node i
7:     **else if** There exists $h' > h$ such that $h' \in h_j^{(i)}$, **then**
8:       $t_{\text{wake}} \leftarrow t_{\text{wake}} + t_I \cdot (h' - h)$
9:       Go to Line 3
10:     **end if**
11:   **end if**
12: **end loop**

D. Optimal Anycast Policy for Periodic Wake-Up Processes

So far, we have developed the value-iteration algorithm and a truncated version of the local-opt algorithm, which are asymptotically optimal for a general sleep-wake scheduling policy. In this subsection, we show that for periodic wake-up patterns these algorithms are exactly optimal for appropriately chosen parameters $\bar{h}$ and $k_{\max}$. In the next section, we will then study why the periodic wake-up pattern is delay-optimal over all the other wake-up patterns.

Assume that all nodes wake up periodically ($\vec{w} = \vec{w}_{\text{per}}$). Then, each neighboring node j must wake up every $1/r_j$ time. Hence, node j must be awake after stage $\lfloor \frac{1/r_j}{t_I} \rfloor$. If we set $\bar{h}$ to the beacon signal $\lfloor \frac{1/r_1}{t_I} \rfloor$, state $x_{\bar{h}} = 1$ is the only admissible state at stage $\bar{h}$. Then, under the periodic wake-up pattern, the result of Proposition 1 becomes stronger as follows:

***Proposition** 3:* If all neighboring nodes wake up periodically, and $\bar{h}$ is set to $\lfloor \frac{1/r_1}{t_I} \rfloor$, the truncated forwarding policy $\hat{f}_i$ is optimal, i.e.,

$$\hat{d}^{(0)}(N_i + 1) = d^{(0)}(N_i + 1).$$

Proof: Since $x_{\bar{h}} = 1$ is the only admissible state, it holds that $\hat{d}^{(\bar{h})}(x_{\bar{h}}) = d^{(\bar{h})}(x_{\bar{h}})$ for admissible states $x_{\bar{h}}$. Then, from (10) and (16), we also have $\hat{d}^{(\bar{h}-1)}_{\text{wait}}(x_{\bar{h}-1}) = d^{(\bar{h}-1)}_{\text{wait}}(x_{\bar{h}-1})$ for all admissible states $x_{\bar{h}-1}$. ($P^{(h)}_{x',x''} = 0$ for inadmissible state $x_h = x''$.) From (12) and (17), it follows that $\hat{d}^{(\bar{h}-1)}(x_{\bar{h}-1}) = d^{(\bar{h}-1)}(x_{\bar{h}} - 1)$ for all admissible states $x_{\bar{h}-1}$. By induction, we can conclude that $\hat{d}^{(0)}(N_i + 1) = d^{(0)}(N_i + 1)$. ■

Proposition 3 implies that the truncated forwarding policy becomes exactly optimal under the periodic wake-up pattern. Hence, when the wake-up pattern of neighboring nodes are periodic, i.e., $\vec{w}_i = \vec{w}_{\text{per}}$, we can completely solve the sub-problem in (5).

The periodic wake-up pattern not only makes the truncated policy optimal, but also simplifies the implementation by the following proposition.

***Proposition** 4:* If all neighboring nodes wake up periodically, and $\bar{h}$ is set to $\lfloor \frac{1/r_1}{t_I} \rfloor$, the conditional delay $\hat{d}^{(h)}_{\text{wait}}(x_h)$ is non-increasing, i.e.,

$$\hat{d}^{(h-1)}_{\text{wait}}(x_{h-1}) \geq \hat{d}^{(h)}_{\text{wait}}(x_h), \tag{27}$$

for $h = 1, 2, \cdots, \bar{h} - 1$, and all admissible states x_h.

The detailed proof is provided in Appendix A. The result of Proposition 4 can be interpreted as follows: as more stages pass by, the neighboring nodes are more likely to wake up, and the conditional delay $d^{(h)}_{\text{wait}}$ then decreases. This property can further simplify the implementation of our solution. Recall that in the original truncated policy, if node j wakes up at beacon signal h and satisfies the condition $D_j + t_D \geq \hat{d}^{(h)}_{\text{wait}}(j)$, it has to sleep and wake up again at the next beacon signal when the condition is satisfied. However, under the periodic wake-up pattern, such a node j will never satisfiy the condition in the following beacon signals because $\hat{d}^{(h)}_{\text{wait}}(j)$ is non-increasing. *Hence, instead of maintaining the set of $h^{(i)}_j$ of all beacon signals that it has to respond with a CTS, each neighboring node j only needs to maintain the last beacon signal that it has to respond.* Furthermore, this property provides an opportunity to reduce the complexity of the LOCAL-OPT algorithm. In [21], we provide the simplified LOCAL-OPT algorithm for the periodic wake-up pattern, whose complexity is reduced $\mathcal{O}(\hat{h}N_i^2)$ to $\mathcal{O}(\hat{h}N_i)$.

We now study the convergence properties of the value-iteration algorithm under the periodic wake-up pattern. Define $\bar{h}^{(k)}_i = \min_{j \in \mathcal{N}_i} \{ \lfloor \frac{1/r_j}{t_I} \rfloor | j \in \arg\min D^{(k-1)}_j \}$ as the maximum number of beacon signals until the neighboring node j with the smallest delay value $D^{(k-1)}_j$ wakes up. Then, the next proposition states the convergence of the value-iteration algorithm:

***Proposition** 5:* If all nodes i wake up periodically and set $\bar{h} = \bar{h}^{(k)}_i$ at each iteration k of the value-iteration algorithm, the algorithm converges to the optimal solution within N iterations, i.e., $D^{(N)}_i = D_i(\vec{r}, \vec{w}_{\text{per}}, f^{(N)}) = D^*_i(\vec{r}, \vec{w}_{\text{per}})$.

Proof: To show the convergence within N iterations, we first show that there exists an acyclic optimal solution, which minimizes the delays from all nodes simultaneously for given sleep-wake scheduling policy $(\vec{r}, \vec{w}_{\text{per}})$, and does not incur any cyclic routing paths. Let f denote an optimal solution. Then, this optimal policy must satisfy the Bellman equation in (4). Hence, for each node i, f_i must minimize the R.H.S. of the sub-problem (5), when the delays of other nodes are given by $D^{(k-1)}_j = D^*_j(\vec{r}, \vec{w}_{\text{per}})$. $D^*_i(\vec{r}, \vec{w}_{\text{per}})$ must be the corresponding delay value $D^{(k)}_i$ in (5). In the sub-problem, to be an eligible next-hop node under the optimal policy f, the neighboring nodes j must satisfy $D^*_j(\vec{r}, \vec{w}) + t_D \leq \hat{d}^{(h)}_{\text{wait}}(j)$ for some h. By repeatedly applying Proposition 4, we have $\hat{d}^{(0)}_{\text{wait}}(N_i + 1) \geq \hat{d}^{(h)}_{\text{wait}}(j)$. Hence, all eligible next-hop nodes j of node i must satisfy $\hat{d}^{(0)}_{\text{wait}}(N_i + 1) \geq D^*_j(\vec{r}, \vec{w}) + t_D$. Since $\hat{d}^{(0)}_{\text{wait}}(N_i + 1)$ corresponds to $D^*_i(\vec{r}, \vec{w})$, we have $D^*_j(\vec{r}, \vec{w}) < D^*_i(\vec{r}, \vec{w})$. This implies that under policy f, a packet at a node i will only be forwarded to a node j, whose delay value $D^*_j(\vec{r}, \vec{w})$ is smaller than $D^*_i(\vec{r}, \vec{w})$. Hence, the solution does not incur any cyclic path.

We have shown the existence of an acyclic solution. Then, based on the proof in [22, Page 107], $D^{(h)}_i$ converges to $D^*_i(\vec{r}, \vec{w})$ for each i within N iterations, and $f^{(N)}$ becomes the corresponding optimal forwarding policy. ■

From Proposition 5, every node needs to run the LOCAL-OPT algorithm for only N iterations, and the last forwarding policy $f^{(N)}$ is delay-optimal when all nodes wake up periodically. Hence, under the periodic wake-up pattern, the overall complexity experienced by each node i is alleviated from $O(k_{\max}\bar{h}N_i^2)$ to $O(N\bar{h}N_i)$. We remind the reader again that this computation overhead only occurs at the configuration phase.

The value-iteration algorithm is a synchronous algorithm that requires all nodes to execute the value-iteration (5) in locked steps. Depending on the application setting, the following asynchronous version of the value-iteration algorithm may be more useful: each node chooses either to solve (5) or to skip it (i.e., $D^{(k)}_i = D^{(k-1)}_i$) independently, of other nodes. Then, the following proposition states the convergence of the asynchronous value-iteration algorithm.

***Proposition** 6:* If each node i updates its delay value $D^{(k)}_i$ using (5) infinitely often, then the delay values and the forwarding policies of all nodes converge to the optimal, i.e., $\lim_{k\to\infty} D^{(k)}_i = D^*_i(\vec{r}, \vec{w}_{\text{per}})$, and $\lim_{k\to\infty} f^{(k)} \in \arg\min_f D_i(\vec{r}, \vec{w}_{\text{per}}, f)$ for all nodes i

Proof: The proof follows from the standard result of Proposition 1.3.5 in [22]. ■

IV. Optimal Wake-up Pattern

In the previous section, we have developed an asymptotically optimal anycast forwarding policy for a general sleep-wake policy $(\vec{r}, \vec{w})$. In this section, we fix the wake-up pattern to the periodic ($\vec{w} = \vec{w}_{\text{per}}$), i.e., all nodes wake up periodically, and study the special properties of the periodic wake-up pattern. We will show that, among all wake-up patterns, the periodic wake-up pattern and the corresponding optimal forwarding policy attain the smallest delay. Hence, they are the solution to the delay-minimization problem (2) that we originally intend to solve.

A. Fundamental Properties of Wake-up Patterns

We begin by studying the fundamental properties of the wake-up patterns. As in Section III-B, we fix the sending node i and a neighboring node j. We define the residual time $R_j(t)$ as the interval from time t to the next wake-up time of node j, *i.e.*, $R_j(t) = \inf_{s:\#_j(s)-\#_j(t)=1} s - t$. Since the wake-up process of a node is a stationary and ergodic process from the viewpoint of other nodes, the distribution of $R_j(t)$ does not depend on time t. Hence, we can drop the variable t and use the random variable R_j to denote the residual time. Let F_{R_j} be the cumulative distribution function (CDF) of R_j, i.e., $F_{R_j}(y) = \Pr(R_j \leq y)$. Note that since nodes wake up independently of other nodes under asynchronous sleep-wake scheduling, the residual time R_j is independent of those of other nodes. Furthermore, since the wake-up process is ergodic, it must satisfy

$$\lim_{T\to\infty} \frac{1}{T}\int_0^T 1_{\{R_j(t)\leq y\}}dt = F_{R_j}(y) \text{ almost surely,} \tag{28}$$

where $1_{\{\cdot\}}$ is an indicator function.

Let $F^*_{R_j}(y)$ be the cumulative distribution function (CDF) of the residual time R_j when the given node j uses the periodic wake-up patterns. Since the node wakes up every $\frac{1}{r_j}$ time in a periodic wake-up process with a random offset, the residual time R_j is uniformly distributed in $[0, \frac{1}{r_j}]$. Hence, the *cdf* of the residual time under the periodic wake-up process is $F^*_{R_j}(y) \triangleq r_j y 1_{\{0\leq y\leq \frac{1}{r_j}\}} + 1_{\{y > \frac{1}{r_j}\}}$. The following proposition then shows the essential properties of the *cdf* of the residual time.

***Proposition** 7:* For any stationary and ergodic wake-up process with rate r_j, the *cdf* $F_{R_j}(y)$ of the residual time R_j satisfies the following properties:

(a) $F_{R_j}(y) \leq F^*_{R_j}(y)$,

(b) $\frac{dF_{R_j}(y)}{dy} \leq \frac{dF^*_{R_j}(y)}{dy}$ for $0 \leq y \leq \frac{1}{r_j}$.

Proof: We first show Property (a). We first estimate

$$\int_0^T 1_{\{R_j(t)<y\}}dt. \tag{29}$$

Let $t_1, t_2, \cdots$ be the sequence of times the node wakes up (as shown in Fig. 2.) To satisfy $R_j(t) < y$, time $t \in [0, T]$ must be in the shaded area, *i.e.*,

$$t \in \cup_{k=1}^{\infty}[t_k - y, t_k]. \tag{30}$$

Hence, we can express (29) as follows:

$$\int_0^T 1_{\{R_j(t)<y\}}dt \leq \sum_{k=1}^{\infty}\int_0^T 1_{\{t\in[t_k-y,t_k]\}}dt \tag{31}$$

$$\leq \sum_{k=1}^{\infty} y 1_{\{t_k\in[0,T+y]\}} = y\cdot\#_j(T+y). \tag{32}$$

From (28), we have $F_{R_j}(y) \leq y \lim_{T\to\infty}\frac{\#_j(T+y)}{T}$. Using (1), we have $F_{R_j}(y) \leq y r_j$ almost surely. This establishes Property (a).

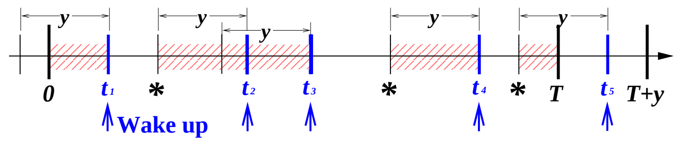


Fig. 2. Example of the sequence of times a node wakes up

We next show Property (b). To show this, we need to compute $F_{R_j}(y_2) - F_{R_j}(y_1)$ for $0 \leq y_1 < y_2 \leq \frac{1}{r_j}$. As we did to show Property (a), we first estimate $\int_0^T 1_{\{R_j(t)\in[y_1,y_2]\}}dt$. We follow the same logic used for showing Property (a). We replace $R_j(t) < y$ and $[t_k - y, t_k]$ with $R_j(t) \in [y_1, y_2]$ and $[t_k - y_2, t_k - y_1]$, respectively, in (29)-(31). Then, the right-hand side of (32) is replaced with $(y_2 - y_1)\cdot\#_j(T + y_2)$. From (28), we have $F_{R_j}(y_2) - F_{R_j}(y_1) \leq (y_2 - y_1)\lim_{T\to\infty}\frac{\#_j(T+y_2)}{T}$. Using (1) again, we have $F_{R_j}(y_2) - F_{R_j}(y_1) \leq r_j(y_2 - y_1)$, which corresponds to Property (b). ■

Proposition 7 shows that for all $0 \leq y \leq 1/r_j$, the *cdf* $F_{R_j}(y)$ and the derivative $\frac{dF_{R_j}(y)}{dy}$ are maximized when the wake-up pattern is periodic. [3]

Recall that the awake probability $p_{j,h}$ is defined as the conditional awake probability that the given node j wakes up and receives the h-th beacon-ID signal, conditioned on that it has not woken up at earlier beacon-ID signals. In order to receive the ID signal h, the residual time R_j until node j wakes up must be in the interval $[(h-1)t_I, ht_I]$, i.e., $p_{j,h} = \Pr\{R_j \in ((h-1)t_I, ht_I] \mid R_j \in [0, (h-1)t_I)\}$. Using the cdf $F_{R_j}(y)$, we can express the awake probability as

$$p_{j,h} = \frac{F_{R_j}(ht_I) - F_{R_j}((h-1)t_I)}{1 - F_{R_j}((h-1)t_I)}. \tag{33}$$

Let $p^*_{j,h}$ be the awake probability of node j when node j wakes up periodically. Since node j wakes up every $1/r_j$ time, the probability $p^*_{j,h}$ is 1 if $h = \lfloor\frac{1/r_j}{t_I}\rfloor$ and

$$p^*_{j,h} = \frac{t_I}{1/r_j - (h-1)t_I} \text{ if } h < \lfloor\frac{1/r_j}{t_I}\rfloor. \tag{34}$$

Then, from Proposition 7, we obtain the following two important properties of wake-up processes.

***Proposition** 8:* For $h = 1, \cdots, \lfloor\frac{1/r_j}{t_I}\rfloor$, we have **(a)** $p^*_{j,h-1} < p^*_{j,h}$, and **(b)** $p^*_{j,h} \geq p_{j,h}$.

Proof: (a) Since node j must wake up by stage $\lfloor\frac{1/r_j}{t_I}\rfloor$, the awake probability is one for $h = \lfloor\frac{1/r_j}{t_I}\rfloor$. Hence, Property (a) holds for this case. For $h < \lfloor\frac{1/r_j}{t_I}\rfloor$, the numerator in (33) is a constant, and the denominator decreases with h. Hence, Property (a) still holds.

[3] Proposition 7 is closely related to the standard results for renewal processes that periodic renewal processes have the smallest mean residual time [23, Chapter 5.2]. These standard results require the wake-up intervals to be independent, while Proposition 7 does not require such an assumption. Since we were unable to find a result in the literature that covered the non-independent case, we have provided the full proof here.

(b) By Proposition 7(a), the denominator is minimized under the periodic wake-up pattern. Further, by Proposition 7(b), the numerator is maximized under the periodic wake-up pattern. Hence, we directly obtain Property (b). ■

Property (a) implies that under the periodic wake-up pattern, the awake probability $p^*_{j,h}$ increases with respect to the number h of the beacon-ID signals sent. Property (b) implies that the conditional awake probability is maximized when the neighboring node wakes up periodically.

B. Optimality of Periodic Wake-up Patterns

Using the properties of the periodic wake-up patterns, we show that the periodic wake-up patterns result in the smallest delay from all nodes. To show this, we first revisit the subproblem (5) that we have solved in Section III-B and in Section III-D.

Consider two scenarios:

(Scenario 1) Each neighboring node j wakes up periodically every $1/r_j$ time. The optimal forwarding policy f_i^* that we obtained in Section III-D is applied. For this scenario, we use the same notations that are used for the optimal forwarding policy, e.g, $d^{(h)}_{\text{wait}}(x_h)$, $d^{(h)}(x_h)$, $P^{(h)}_{x_{h-1},x_h}$, $h_{j,\max}$, and $x_{h,\max}$. Recall that the packet at the sending node is forwarded no later than stage $\bar{h} = h_{1,\max}$.

(Scenario 2) The wake-up process of each neighboring node j is arbitrary, but the wake-up rate is still given by r_j. We denote by $\tilde{f}_i$ the optimal forwarding policy for the given wake-up processes of the neighboring nodes. To differentiate from Scenario 1, we put a tilde ($\sim$) on all notations in this scenario, *e.g.*, $\tilde{d}^{(h)}_{\text{wait}}(x_h)$, $\tilde{d}^{(h)}(x_h)$, $\tilde{P}^{(h)}_{x_{h-1},x_h}$, etc. Similarly, node j must have woken up no later than stage $\tilde{h}_{j,\max}$, and let $\tilde{x}_{h,\max}$ be the node with the smallest delay among the nodes that must be awake at stage h. By simply setting $\tilde{h}_{j,\max} = \infty$ and $\tilde{x}_{h,\max} = N_i + 1$, we can still use these notations for the wake-up processes under which there is no such a finite limit point. For instance, if all neighboring nodes j follow the Poisson wake-up pattern, then the residual times until they wake up are independent exponential random variables, and we thus have $\tilde{h}_{j,\max} = \infty$ for $j \in \mathcal{N}_i$ and $\tilde{x}_{h,\max} = N_i + 1$ for all $h \geq 0$. Since the awake probability is maximized when nodes wake up periodically, it follows that $h_{j,\max} \leq \tilde{h}_{j,\max}$ and $x_{h,\max} \leq \tilde{x}_{h,\max}$. Further, the optimal policy $\tilde{f}_i$ must satisfy the necessary conditions (12) and (11).

We now compare the delays from both scenarios.

***Proposition** 9:* $d^{(h)}(x_h) \leq \tilde{d}^{(h)}(x_h)$ for $h = 0, 1, \cdots, \bar{h}$ and $x_h \leq x_{h,\max}$,

Proof: We prove this by induction. By (13), we must have $d^{(\bar{h})}(1) = \tilde{d}^{(\bar{h})}(1) = t_I + t_D + D_1$. At stage $\bar{h}$, node 1 must be awake under the periodic wake-up process (i.e., $x_{\bar{h},\max} = 1$). Hence, Proposition 9 holds for $h = \bar{h}$.

Assume that $d^{(h)}(x_h) \leq \tilde{d}^{(h)}(x_h)$ holds for $h = h'+1, h'+2, \cdots, \bar{h}$ and $x_h \leq x_{h,\max}$. We then show that this also holds for $h = h'$. From (10), we have the following inequality:

$$\begin{aligned}
&\tilde{d}^{(h')}_{\text{wait}}(x_{h'}) - t_I = \textstyle\sum_{x_{h'+1}=1}^{x_{h'}} \tilde{P}^{(h'+1)}_{x_{h'},x_{h'+1}} \tilde{d}^{(h'+1)}(x_{h'+1}) \\
&\geq \textstyle\sum_{x_{h'+1}=1}^{x_{h'}} \tilde{P}^{(h'+1)}_{x_{h'},x_{h'+1}} d^{(h'+1)}(x_{h'+1}) \qquad (35)\\
&\geq \textstyle\sum_{x_{h'+1}=1}^{x_{h'}} P^{(h'+1)}_{x_{h'},x_{h'+1}} d^{(h'+1)}(x_{h'+1}) \qquad (36)
\end{aligned}$$

To obtain (35), we have used the induction hypothesis. The inequality in (36) can be understood as follows: according to Proposition 8(b), neighboring nodes are more likely to wake up under the periodic wake-up patterns, and thus the delay is also minimized under the periodic wake-up pattern. To obtain (36), we have used Lemma 1 in Appendix A, where $L = x_{h'}$, $\alpha_j^{(1)} = p'_{j,h'+1}$ (equivalently, $\beta_j^{(1)} = \tilde{P}^{(h'+1)}_{x_{h'},j}$), $\alpha_j^{(2)} = p_{j,h'+1}$ (equivalently, $\beta_j^{(2)} = P^{(h'+1)}_{x_{h'},j}$), and $\theta_j = d^{(h'+1)}(j)$. Since $\theta_1 \leq \cdots \leq \theta_{\bar{h}}$ by Proposition 24, and $\alpha_j^{(1)} \leq \alpha_j^{(2)}$ by Proposition 8, the conditions for the lemma hold.

Since (36) is equal to $d^{(h')}_{\text{wait}}(x_{h'}) - t_I$, we have $d^{(h')}_{\text{wait}}(x_{h'}) \leq \tilde{d}^{(h')}_{\text{wait}}(x_{h'})$. Then, from (12), we have $d^{(h')}(x_{h'}) \leq \tilde{d}^{(h')}(x_{h'})$. Hence, Proposition 9 holds for $h = h'$. By induction, this also holds for $h = 0, 1, \cdots, \bar{h}$. ■

From Proposition 9, we can infer that $d^{(0)}(N_i + 1) \leq \tilde{d}^{(0)}(N_i + 1)$, which implies $D_i^{(k)} \leq \tilde{D}_i^{(k)}$ in the value iteration algorithm. Hence, when the delays from the neighboring nodes are given, the delay from the sending node i is minimized when the neighboring nodes wake up periodically and the corresponding optimal forwarding policy is applied.

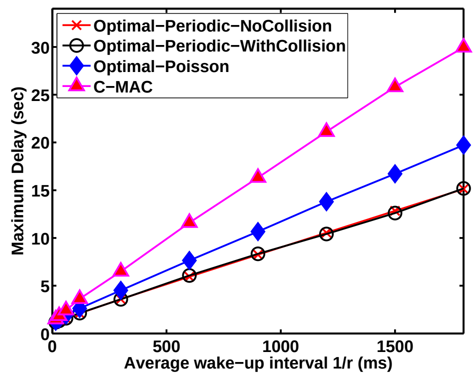
We next apply this result to the Stochastic Shortest Path (SSP) problem in (3). Assume that each node i can control the wake-up patterns $\vec{w}_i$ of its neighboring nodes j, as well as its forwarding policy f_i. Then, to minimize (3) with respect to $(\vec{w}, f)$, every node i should carry out the following value-iteration algorithm, which is a generalized version of (5): for $k = 1, 2, \cdots$, $D_i^{(k)} = \min_{\vec{w}_i, f_i}(D_{\text{hop},i}(\vec{r}, \vec{w}_i, f_i) + \sum_{j \in \mathcal{N}_i} q_{i,j}(\vec{r}, \vec{w}_i, f_i) D_j^{(k-1)})$. In this equation, the expected one-hop delay $D_{\text{hop},i}(\vec{r}, \vec{w}_i, f_i)$ and the probability $q_{i,j}(\vec{r}, \vec{w}_i, f_i)$ that node i forwards the packet to node j depend only on $\vec{w}_i$ (instead of $\vec{w}$). This is because the wake-up patterns of nodes other than the neighboring nodes do not affect the one-hop delay and the transition probability from node i. From Proposition 9, $D_i^{(k)}$ is maximized when $\vec{w}_i$ is given by $\vec{w}_{\text{per}}$ and the corresponding optimal forwarding policy is chosen. Hence, the following proposition holds:

***Proposition** 10:* $\min_f D_i(\vec{r}, \vec{w}_{\text{per}}, f) = \min_{\vec{w}, f} D_i(\vec{r}, \vec{w}, f)$ for all nodes i.

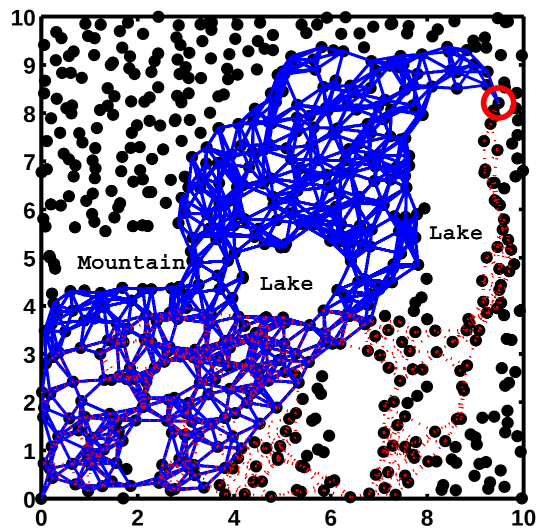
Let $f^*(\vec{r})$ be the optimal forwarding policy for a given sleep-wake scheduling policy $(\vec{r}, \vec{w}_{\text{per}})$. From Proposition 5, $f^*(\vec{r})$ is equal to $f^{(N)}$ in the value-iteration algorithm. Then, Proposition 10 implies that $(\vec{w}_{\text{per}}, f^*(\vec{r}))$ is the solution to the delay-minimization problem (2).

V. Simulation Results

In this section, we provide simulation results to evaluate the delay performance of the proposed solution. To simulate more realistic scenarios, we randomly deploy 690 nodes over a 1 km-by-1km area with obstructions as shown in Fig 3(b).



(a) Delay Comparison



(b) Node Deployment and Possible Paths

Fig. 3. (a) Maximum delay under different wake-up rate r and (b) Node deployment and the possible routing paths for 300 ms average wake-up interval under Optimal-Periodic-NoCollision (blue solid lines) and CMAC (red dotted lines). The paths under Optimal-Periodic-NoCollision pass through the network diagonally while the paths under CMAC circumvent the lake.

We set the transmission range to 70 m and the duration t_I and t_D to 6 ms and 30 ms, respectively.

We will compare the delay performance of the following algorithms:

Optimal-Periodic-NoCollision: This corresponds to the optimal anycast forwarding policy with periodic wake-up patterns, and the effect of collision is ignored. We obtain the expected delay simply from the output of the value iteration algorithm in (5).

Optimal-Periodic-WithCollision: This corresponds to the optimal anycast policy with periodic wake-up patterns. We simulate the policy with the collision resolution component in [21, Appendix D, Deterministic Backoff]

Optimal-Poisson: This corresponds to the optimal anycast forwarding policy in [20] with Poisson wake-up patterns. We also simulate the policy with the same collision resolution component in Optimal-Periodic-WithCollision. (Refer to [20] to see the performance advantage of **Optimal-Poisson** over existing solutions (includeling CMAC) over different simulation environments.)

CMAC (Convergent MAC): This corresponds to the heuristic algorithm with Poisson wake-up pattern that was proposed in [12]. CMAC uses geographical information to choose the packet forwarding policy. Let D and R be the random variables that denote the one-hop delay and process in reducing the Euclidean distance to the sink when a packet is forwarded to the next-hop node. Then, under CMAC, each node i selects the set of eligible next-hop nodes that can maximize the expected normalized-latency $E[D/R]$. Since the performance advantage of CMAC over other existing anycast-based heuristics has been extensively studied in [12] and [20], we only compare the performance of our optimal algorithm to that of CMAC.
To simulate these algorithms, we generate 50 packets at each node and take the average on the measured delay.

In Fig. 3(a), we compare the maximum expected end-to-end delay over all nodes under different wake-up rates r. We observe that ‘Optimal-Periodic-NoCollision’ and ‘Optimal-Periodic-WithCollision’ significantly reduce the end-to-end delay compared with the other algorithms. This is consistent with our result that the periodic wake-up pattern is delay-optimal. We also observe the significant performance gap between ‘CMAC’ and ‘Optimal-Periodic-WithCollision.’ To explain this performance gap, we show in Fig. 3(b) the possible routing paths under both algorithms. Under CMAC, packets tend to be forwarded to the nodes with higher progress. However, overall the packets may take longer paths to go around the obstructions. In contrast, under ‘Optimal-Periodic-Withcollision,’ the next-hop nodes are chosen by delay. Hence, it is possible for a packet to be first forwarded to nodes with negative progress, if doing so reduces the delay beyond the next-hop node. For example, in Fig. 3(b), ‘Optimal-Periodic-WithCollision’ results in paths that are shorter than those under ‘CMAC.’ From Fig. 3(b), we can infer that if there is no strong correlation between distance and delay (e.g. where there are obstructions), the heuristic anycast solutions such as CMAC can perform poorly. Finally, we can observe from Fig. 3(a) that the performance gap between ‘Optimal-Periodic-NoCollision’ and ‘Optimal-Periodic-WithCollision’ is negligible over average wake-up intervals (from 30 ms to 1800 ms). Hence, as long as collisions are resolved properly, they will not significantly impact the performance of our proposed solution at reasonable wake-up rates.

VI. Conclusion

In this paper, we have studied the optimal anycast forwarding and sleep-wake scheduling policies that minimize the end-to-end delay. We have shown that among all wake-up patterns with the same wake-up rate, the periodic wake-up pattern maximizes the probability that a neighboring node wakes up at each beacon signal. Using this result, we have developed the optimal anycast forwarding algorithms for periodic wake-up patterns and have shown that the algorithms guarantee the minimum end-to-end delay of all nodes for given wake-up rates (which correspond to given energy budgets). Through simulation results, we have illustrated the benefits of using asynchronous periodic sleep-wake scheduling.

Appendix A
Proof of Proposition 4

Proof: We prove by induction that $\hat{d}_{\text{wait}}^{(h-1)}(x) \geq \hat{d}_{\text{wait}}^{(h)}(x)$ holds for $h = \bar{h}-1, \cdots, 1, 0$ and admissible states $x_h = x$. At stage $\bar{h}-1$, from (16), we have $\hat{d}_{\text{wait}}^{(\bar{h}-1)}(x) = t_I + \hat{d}^{(\bar{h})}(1)$ because state 1 is the only admissible state at stage $\bar{h}$. By (17), we have $\hat{d}_{\text{wait}}^{(\bar{h}-1)}(x) = t_I + t_D + D_1$. At stage $\bar{h}-2$, we must have $\hat{d}_{\text{wait}}^{(\bar{h}-2)}(x) \geq t_I + t_D + D_1$ since $\hat{d}^{(\bar{h}-1)}(x_{\bar{h}-1}) \geq t_D + D_1$. Thus, it holds that $\hat{d}_{\text{wait}}^{(\bar{h}-2)}(x) \leq \hat{d}_{\text{wait}}^{(\bar{h}-1)}(x)$ for admissible states $x_{\bar{h}-1} = x$.

Assume that $d_{\text{wait}}^{(h-1)}(x) \geq d_{\text{wait}}^{(h)}(x)$ holds for $h = h'+1, h'+2, \cdots, \bar{h}$ and admissible states $x_h = x$. We then show that this also holds for $h = h'$. To this end, we need the following lemma:

***Lemma** 1:* Suppose $\alpha_j^{(1)}$, $\alpha_j^{(2)}$, $\beta_j^{(1)}$, $\beta_j^{(2)}$, and θ_j for $j = 1, \cdots, L$ such that $0 \leq \alpha_j^{(1)} \leq \alpha_j^{(2)} \leq 1$, $\alpha_L^{(m)} = 1$, $\beta_j^{(m)} = \prod_{k=1}^{j-1}(1-\alpha_k^{(m)})\alpha_j^{(m)}$ for $m = 1, 2$, and $\theta_1 \leq \theta_2 \leq \cdots \leq \theta_L$. Then, the following inequality holds:

$$\sum_{j=1}^{L} \beta_j^{(1)}\theta_j \geq \sum_{j=1}^{L} \beta_j^{(2)}\theta_j. \tag{37}$$

The detailed proof is provided in Appendix B. Lemma 1 has the following interpretation. Assume that there are two users $m = 1, 2$ and each user m picks up at least one θ_j's from $\{\theta_1, \theta_2, \cdots, \theta_L\}$ independently of the other user. $\alpha_j^{(m)}$ is the probability that user m will pick θ_j, independently of whether it picks other θ_k's ($k \neq j$). Since $\alpha_L^{(m)} = 1$, at least θ_L must be picked up by each user. If $\theta_1 \leq \theta_2 \leq \cdots \leq \theta_L$, then for the user with a larger value of $\alpha_j^{(m)}$, the expected value of the *smallest* θ_j picked will be lower.

Using Lemma 1, we can show the following inequality: for all $x \leq x_{h',\max}$

$$\begin{aligned} d_{\text{wait}}^{(h'-1)}(x) =& t_I + \sum_{x'=1}^{x} P_{x,x'}^{(h')} d^{(h')}(x') \\ \geq& t_I + \sum_{x'=1}^{x} P_{x,x'}^{(h'+1)} d^{(h')}(x'). \end{aligned} \tag{38}$$

Let L in Lemma 1 be x. For $m = 1, 2$, let $\alpha_j^{(m)} = p_{j,h'-1+m}$ if $1 \leq j < L$, and let $\alpha_L^{(m)} = 1$. Since $\beta_j^{(m)} = \prod_{h'=1}^{j-1}(1-\alpha_{h'}^{(m)})\alpha_j^{(m)}$, $\beta_j^{(m)}$ is given by $P_{x,j}^{(h'-1+m)}$ from (6). Note that under the periodic wake-up process, the awake probability $p_{j,h}$ in (34) increases with h, which means $0 \leq \alpha_j^{(1)} \leq \alpha_j^{(2)} \leq 1$. Let $\theta_j = d^{(h')}(j)$. By Proposition 24, we have $d^{(h')}(1) \leq d^{(h')}(2) \leq \cdots \leq d^{(h')}(x_{h',\max})$, which satisfies the condition $\theta_1 \leq \theta_2 \leq \cdots \leq \theta_L$. Since all conditions for $\alpha_j^{(m)}$, $\beta_j^{(m)}$, and θ_j ($j = 1, 2, \cdots, L$ and $m = 1, 2$) are satisfied, we obtain the inequality in (38) from Lemma 1.

Combining the induction hypothesis and (12), we can obtain $d^{(h')}(x') \geq d^{(h'+1)}(x')$ for $1 \leq x' \leq x_{h'+1,\max}$. Since $P_{x,x'}^{(h'+1)} = 0$ for $x' > x_{h'+1,\max}$, we can obtain the following inequality from (38): for $x \leq x_{h',\max}$,

$$\begin{aligned} & t_I + \sum_{x'=1}^{x} P_{x,x'}^{(h'+1)} d^{(h')}(x') \\ & \geq t_I + \sum_{x'=1}^{x} P_{x,x'}^{(h'+1)} d^{(h'+1)}(x') \\ & = d_{\text{wait}}^{(h')}(x). \end{aligned} \tag{39}$$

Combining (38) and (39), we have $d_{\text{wait}}^{(h'-1)}(x) \geq d_{\text{wait}}^{(h')}(x)$ for $1 \leq x \leq x_{h',\max}$. By induction, Proposition 4 follows. ■

Appendix B
Proof of Lemma 1

Proof: We prove this lemma by induction. First, the lemma holds for $L = 1$ because $\alpha_1^{(1)} = \alpha_1^{(2)} = 1$ and

$$\beta_1^{(1)}\theta_1 = \alpha_1^{(1)}\theta_1 = \alpha_1^{(2)}\theta_1 = \beta_1^{(2)}\theta_1.$$

We now assume that (37) holds for $L = 1, 2, \cdots, K-1$ and suppose $L = K$. Let $\tilde{\alpha}_j^{(m)} \triangleq \alpha_{j+1}^{(m)}$, $\tilde{\theta}_j \triangleq \theta_{j+1}$, and

$$\tilde{\beta}_j^{(m)} \triangleq \frac{\beta_{j+1}^{(m)}}{1-\alpha_1^{(m)}} = \prod_{k=1}^{j-1}(1-\tilde{\alpha}_k^{(m)})\tilde{\alpha}_j^{(m)}$$

for $m = 1, 2$ and $j = 1, 2, \cdots, K-1$. Then, by induction hypothesis, we have

$$\sum_{j=1}^{K-1} \tilde{\beta}_j^{(1)}\tilde{\theta}_j \geq \sum_{j=1}^{K-1} \tilde{\beta}_j^{(2)}\tilde{\theta}_j. \tag{40}$$

Using the above, we can obtain the following inequality:

$$\begin{aligned} \sum_{j=1}^{K} \beta_j^{(1)}\theta_j &= \alpha_1^{(1)}\theta_1 + \sum_{j=2}^{K} \beta_j^{(1)}\theta_j \\ &= \alpha_1^{(1)}\theta_1 + (1-\alpha_1^{(1)}) \sum_{j=1}^{K-1} \tilde{\beta}_j^{(1)}\tilde{\theta}_j \\ &\geq \alpha_1^{(1)}\theta_1 + (1-\alpha_1^{(1)}) \sum_{j=1}^{K-1} \tilde{\beta}_j^{(2)}\tilde{\theta}_j \qquad (41) \\ &= \alpha_1^{(1)} \left(\theta_1 - \sum_{j=1}^{K-1} \tilde{\beta}_j^{(2)}\tilde{\theta}_j\right) + \sum_{j=1}^{K-1} \tilde{\beta}_j^{(2)}\tilde{\theta}_j. \qquad (42) \end{aligned}$$

To obtain (41), we have used (40). Since $\sum_{j=1}^{K-1} \tilde{\beta}_j^{(2)}\tilde{\theta}_j$ is a weighted average of $\theta_2, \theta_3, \cdots, \theta_K$, and all these values are no smaller than θ_1, the term $(\theta_1 - \sum_{j=1}^{K-1} \tilde{\beta}_j^{(2)}\tilde{\theta}_j)$ is non-positive. Since $\alpha_1^{(1)} \leq \alpha_1^{(2)}$, we can rewrite (42) as

$$\begin{aligned} \sum_{j=1}^{K} \beta_j^{(1)}\theta_j &\geq \alpha_1^{(2)} \left(\theta_1 - \sum_{j=1}^{K-1} \tilde{\beta}_j^{(2)}\tilde{\theta}_j\right) + \sum_{j=1}^{K-1} \tilde{\beta}_j^{(2)}\tilde{\theta}_j \\ &= \alpha_1^{(2)}\theta_1 + (1-\alpha_1^{(2)}) \sum_{j=1}^{K-1} \tilde{\beta}_j^{(2)}\tilde{\theta}_j. \\ &= \sum_{j=1}^{K} \beta_j^{(2)}\theta_j. \qquad (43) \end{aligned}$$

Hence, (37) holds for $L = K$. By induction, the result of the lemma follows. ■

References

[1] J. Kim, X. Lin, and N. B. Shroff, "Optimal Anycast Technique for Delay-Sensitive Energy-Constrained Asynchronous Sensor Networks," in *Proceedings of IEEE INFOCOM*, (Rio De Janeiro, Brazil), April 2009.

[2] Y.-C. Tseng, C.-S. Hsu, and T.-Y. Hsieh, "Power-Saving Protocols for IEEE 802.11-Based Multi-Hop Ad Hoc Networks," *Computer Networks*, vol. 43, pp. 317–337, Oct. 2003.

[3] W. Ye, H. Heidemann, and D. Estrin, "Medium Access Control with Coordinated Adaptive Sleeping for Wireless Sensor Networks," *IEEE/ACM Transactions on Networking*, vol. 12, pp. 493–506, June 2004.

[4] T. van Dam and K. Langendoen, "An Adaptive Energy-Efficient MAC Protocol for Wireless Sensor Networks," in *Proc. SenSys*, pp. 171–180, November 2003.

[5] G. Lu, B. Krishnamachari, and C. S. Raghavendra, "An Adaptive Energy-Efficient and Low-Latency MAC for Data Gathering in Wireless Sensor Networks," in *Proc. IPDPS*, pp. 224–231, April 2004.

[6] J. Elson, L. Girod, and D. Estrin, "Fine-grained network time synchronization using reference broadcasts," *SIGOPS Oper. Syst. Rev.*, vol. 36, no. SI, pp. 147–163, 2002.

[7] C. Schurgers, V. Tsiatsis, S. Ganeriwal, and M. Srivastava, "Optimizing sensor networks in the energy-latency-density design space," *IEEE Transactions on Mobile Computing*, vol. 1, pp. 70–80, January-March 2002.

[8] J. Polastre, J. Hill, and D. Culler, "Versatile Low Power Media Access for Wireless Sensor Networks," in *Proc. SenSys*, pp. 95–107, November 2004.

[9] J. Polastre, J. Hill, P. Levis, J. Zhao, D. Culler, and S. Shenker, "A Unifying Link Abstraction for Wireless Sensor Networks," in *Proc. SenSys*, pp. 76–89, November 2005.

[10] M. Zorzi and R. R. Rao, "Geographic Random Forwarding (GeRaF) for Ad Hoc and Sensor Networks: Energy and Latency Performance," *IEEE transactions on Mobile Computing*, vol. 2, pp. 349–365, October 2003.

[11] M. Zorzi and R. R. Rao, "Geographic Random Forwarding (GeRaF) for Ad hoc and Sensor Networks: Multihop Performance," *IEEE Transactions on Mobile Computing*, vol. 2, pp. 337–348, October 2003.

[12] S. Liu, K.-W. Fan, and P. Sinha, "CMAC: An Energy Efficient MAC Layer Protocol Using Convergent Packet Forwarding for Wireless Sensor Networks," in *Proc. SECON*, (San Diego, CA), June 2007.

[13] R. R. Choudhury and N. H. Vaidya, "MAC-Layer Anycasting in Ad Hoc Networks," *SIGCOMM Computer Communication Review*, vol. 34, pp. 75–80, January 2004.

[14] S. Jain and S. R. Das, "Exploiting Path Diversity in the Link Layer in Wireless Ad Hoc Networks," in *Proc. WoWMoM*, pp. 22–30, June 2007.

[15] P. Larsson and N. Johansson, "Multiuser diversity forwarding in multihop routing for wireless networks," in *Proceedings of IEEE WCNC*, 2005.

[16] S. Biswas and R. Morris, "ExOR: opportunistic multi-hop routing for wireless networks," *Proceedings of ACM SIGCOMM*, vol. 35, pp. 133–144, October 2005.

[17] M. Rossi and M. Zorzi, "Integrated Cost-Based MAC and Routing Techniques for Hop Count Forwarding in Wireless Sensor Networks," *IEEE Transactions on Mobile Computing*, vol. 6, pp. 434–448, April 2007.

[18] M. Rossi, M. Zorzi, and R. R. Rao, "Statistically Assisted Routing Algorithm (SARA) for Hop Count Based Forwarding in Wireless Sensor Networks," *Wireless Networks*, vol. 14, pp. 55–70, February 2008.

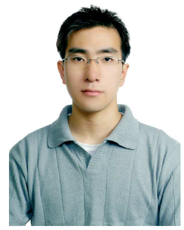
[19] J. Kim, X. Lin, N. B. Shroff, and P. Sinha, "On Maximizing the Lifetime of Delay-Sensitive Wireless Sensor Networks with Anycast," in *Proceedings of IEEE INFOCOM*, (Pheonix, AZ), April 2008.

[20] J. Kim, X. Lin, N. B. Shroff, and P. Sinha, "Minimizing Delay and Maximizing Lifetime for Wireless Sensor Networks with Anycast," *accepted for publication in* IEEE/ACM Trans. on Networking*, 2009, and also available at http://web.ics.purdue.edu/˜kim309*.

[21] J. Kim, X. Lin, and N. B. Shroff, "Optimal Anycast Technique for Delay-Sensitive Energy-Constrained Asynchronous Sensor Networks," *Technical Report, http://web.ics.purdue.edu/˜kim309/Kim08tech3.pdf,* 2008.

[22] D. P. Bertsekas, *Dynamic Programming and Optimal Control vol. 2*. Athena Scientific, 3 ed., 2007.

[23] L. Kleinrock, *Queueing Systems vol. 1: Theory*. Wiley-Interscience, 1 ed., 1975.



Joohwan Kim (S'07) received his B.S. degree from Yonsei University, Seoul, Korea, in 2004, and his M.S. degree from Purdue University, West Lafayette, Indiana in 2006. He is currently a Ph.D. candidate of Electrical and Computer Engineering at Purdue University.

His research interests range over the various area of wireless communication networks: scheduling, routing, power control, network pricing and wireless resource optimization in sensor and mobile ad hoc networks.

Xiaojun Lin (S'02 / M'05) received his B.S. from Zhongshan University, Guangzhou, China, in 1994, and his M.S. and Ph.D. degrees from Purdue University, West Lafayette, Indiana, in 2000 and 2005, respectively. He is currently an Assistant Professor of Electrical and Computer Engineering at Purdue University.

Dr. Lin's research interests are resource allocation, optimization, network pricing, routing, congestion control, network as a large system, cross-layer design in wireless networks, mobile ad hoc and sensor networks.

He received the IEEE INFOCOM 2008 best paper award and 2005 best paper of the year award from *Journal of Communications and Networks*. His paper was also one of two runner-up papers for the best-paper award at IEEE INFOCOM 2005. He received the NSF CAREER award in 2007.

Ness B. Shroff (S'91 / M'93 / SM'01 / F'07) Ness B. Shroff received his Ph.D. degree from Columbia University, NY in 1994 and joined Purdue university immediately thereafter as an Assistant Professor. At Purdue, he became Professor of the school of Electrical and Computer Engineering in 2003 and director of CWSA in 2004, a university-wide center on wireless systems and applications. In 2007, he joined The Ohio State University as the Ohio Eminent Scholar of Networking and Communications and Professor of ECE and CSE.

His research interests span the areas of wireless and wireline communication networks. He is especially interested in fundamental problems in the design, performance, control, and security of these networks.

Dr. Shroff has served on the editorial boards of IEEE/ACM Trans. on Networking, the Computer Networks Journal, and IEEE Communications Letters. He was the technical program co-chair of IEEE INFOCOM'03, IEEE CCW'99, the program co-chair for the symposium on high-speed networks, Globecom 2001, and the panel co-chair for ACM Mobicom'02. He was also a co-organizer of the NSF workshop on Fundamental Research in Networking, held in Arlie House Virginia, in 2003. In 2008, he served as the technical program co-chair of ACM Mobihoc 2008 and as General co-chair of WICON. Dr. Shroff received the IEEE INFOCOM 2008 best paper award, IEEE INFOCOM 2006 best paper award, the IEEE IWQoS 2006 best student paper award, the 2005 best paper of the year award for the Journal of Communications and Networking, the 2003 best paper of the year award for Computer Networks, and the NSF CAREER award in 1996 (his INFOCOM 2005 paper was also selected as one of two runner-up papers for the best paper award).