Hindawi Publishing Corporation
EURASIP Journal on Wireless Communications and Networking
Volume 2006, Article ID 39814, Pages 1–17
DOI 10.1155/WCN/2006/39814

Energy-Efficient Medium Access Control Protocols for Wireless Sensor Networks

Qingchun Ren and Qilian Liang

Department of Electrical Engineering, The University of Texas at Arlington, Arlington, TX 76019-0016, USA

Received 3 November 2005; Revised 14 April 2006; Accepted 2 May 2006

Recommended for Publication by Dongmei Zhao

A key challenge for wireless sensor networks is how to extend network lifetime with dynamic power management on energy-constraint sensor nodes. In this paper, we propose two energy-efficient MAC protocols: asynchronous MAC (A-MAC) protocol and asynchronous schedule-based MAC (ASMAC) protocol. A-MAC and ASMAC protocols are attractive due to their suitabilities for multihop networks and capabilities of removing accumulative clock-drifts without any network synchronization. Moreover, we build a traffic-strength- and network-density-based model to adjust essential algorithm parameters adaptively. Simulation results show that our algorithms can successfully acquire the optimum values of power-on/off duration, schedule-broadcast interval, as well as super-time-slot size and order. These algorithm parameters can ensure adequate successful transmission rate, short waiting time, and high energy utilization. Therefore, not only the performance of network is improved but also its lifetime is extended when A-MAC or ASMAC is used.

1. INTRODUCTION AND MOTIVATIONS

A wireless sensor network (WSN) can be thought as an ad hoc network consisting of sensor nodes that are linked by wireless medium to perform distributed sensing tasks. Recent developments in integrated circuit technology have brought about the construction of small and low-cost sensor node with signal processing and wireless communication capabilities. Distributed WSNs have increasing applications, as they hold the potential to renovate many segments of our economies and lives from environment monitoring to manufacture and business asset management [1].

One crucial challenge for WSN designers is to develop a system that will run for years unattendedly, which calls for not only robust hardware and software, but also lasting energy sources. However, currently, sensor nodes are powered by battery, whose available energy is limited. Moreover, replacing or recharging battery, in many cases, may be impractical or uneconomical. Even though, future sensor nodes may be powered by ambient energy sources (such as sunlight, vibrations, etc.) [2], the provided current is very low. From both perspectives, protocols and applications designed for WSNs should be highly efficient and optimized in terms of energy.

In general, a sensor node consists of a microprocessor, a data storage, sensors, analog-to-digital converters (ADCs), a data transceiver, an energy source, and controllers that tie those pieces together [1]. Communications, not only transmitting, but also receiving, or merely scanning a channel for communication, can use up to half of the energy [3]. Thus, recently, some researchers have begun to study the energy efficiency problem through reducing power consumption on wireless interface.

Commonly, a distributed WSN is composed of a set of low-end data-gathering sensor nodes and high-end data-collection sensor nodes. In kinds of network, data-collection sensor nodes collect the data about a physical phenomenon and send them to related data-gathering sensor nodes that act as lead-sensor or fusion center over wireless links. For example, in [4, 5], such network model was employed for investigating the energy efficiency of distributed coding and signal processing. A similar model was employed in [6] to develop a collaborative and distributed tracking algorithm for energy-aware WSNs.

In a WSN with hierarchical topology, communications can be divided into three main categories based on communication terminals, that is, communication between data-collection nodes, communication between data-gathering

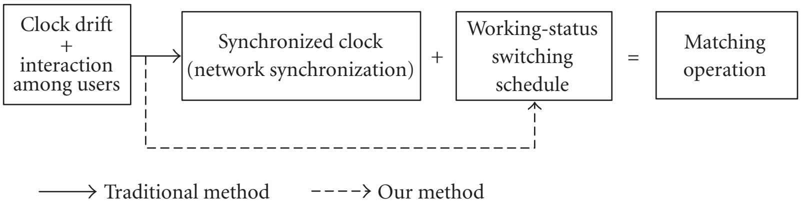

Clock drift + interaction among users
Synchronized clock (network synchronization)
+
Working-status switching schedule
=
Matching operation
Traditional method
Our method

Figure 1: Motivation of our energy-efficient MAC protocols.

nodes, as well as communication between data-collection nodes and data-gathering nodes. In this paper, we mainly focus on how to design energy-efficient MAC protocols to organize the communication between data-collection nodes, which suffer from power constraint strictly. This type of communication is quite common in general WSNs. For instance, data-collection nodes exchange their collected information before sending it to data-gathering nodes to reduce information redundancy caused by position correlation of nodes. Another example is given in [7], in which to implement V-BLAST-based virtual multiple-input multiple-output (MIMO) communication, data-collection nodes share their collected information with each other before transmission.

1.1. Accumulative clock-drift problem

As a matter of fact, the quality of a node's clock usually boils down to its frequency's stability and accuracy [8]. Generally speaking, as frequency stability and accuracy increase, so do its required power, size, and cost, which are all troublesome for general nodes. Moreover, the frequency generated by a quartz oscillator is also affected by a number of environmental factors: voltage applied to it, ambient temperature, acceleration in space, and so forth. Low-cost oscillators commonly have nominal frequency accuracy on the order of 10^4 to 10^6. That is, two similar but uncalibrated oscillators will drift apart from 1 to 100 microseconds every second [8]. As time goes, oscillators will drift apart farther and farther. We call this accumulative clock-drift in this paper.

The basic idea of most energy-efficient MAC protocols is to power on/off their radios alternately to implement communication and to reduce energy consumption. This active/sleep scheme requires matching operation among nodes (i.e., source-destination pairs switch between active and sleep states coincidently) to ensure that the low-power radio schedule works successfully. Hence, for general WSNs, it is necessary to develop effective and efficient methods to resolve the mismatching problem caused by accumulative clock-drift. Network synchronization is one of the existing approaches for this issue, in which a common timescale is necessary. However, is it the only or the best choice? If a system dose not provide network synchronization service, is there any alternative solution? Furthermore, although the strategies exploited by existing network synchronization schemes are various, the working load for carrying out network synchronization is mainly located at the user's sides (or data-collection node sides), which we call user-exhaustion schemes. Obviously, user-exhaustion scheme is not a wise choice, since data-collection nodes are typically subjected to strict energy constraint while data-gathering nodes are not.

1.2. Heterogeneous problem

The traffic of WSN, in general, has a heterogeneous nature [9] (i.e., the traffic arrival rate for different sensor nodes and even for the same sensor node at different time fluctuates considerably during the network lifetime). Consequently, according to the time-variant situation of system, how to adjust essential parameters adaptively is another important task for protocol designers. As a matter of fact, we notice that the power-on/off duration is tightly related to the system performance in terms of energy saving, time delay, and system throughput. That is, with the increase of power-off duration, there is more chance for buffer overflowing, longer waiting-time for data packets, and fewer data packets being transmitted during a period of time. However there is more energy reserved for avoiding excessive idle listening. On the other hand, with the increase of power-on duration, there are more data packets transmitted, then there is less chance for buffer overflowing and shorter waiting-time for data packets. However, there is more energy wasted by idle listening. Nevertheless, little work is done on how to determine those essential parameters.

1.3. Our contributions

Leveraging the characteristics of free-running timing method and the advantages of fuzzy logic system on uncertain problems, we propose two energy-efficient MAC protocols for WSNs: asynchronous MAC (A-MAC) protocol and asynchronous schedule-based MAC (ASMAC) protocol. Our timing-rescheduling scheme and time-slot allocation algorithm provide an approach to remove the tight dependency on network synchronization for energy-efficient MAC protocols, which is a critical constraint for network upgrading and expanding (Figure 1). Within A-MAC and ASMAC protocols, no common timescale is needed any more, which will free the energy for setting up and maintaining.

Furthermore, considering the heterogeneous nature of WSN, we build a traffic-strength- and network-density-based designing model. This model equips the system with the capability to determine essential algorithm parameters adaptively, which greatly influence system performance in terms of energy reservation and communication capability. Those algorithm parameters include power-on/off duration, schedule-broadcast interval, as well as super-time-slot size and order. In addition, static approaches may be far from being optimal because they deny the opportunity to reschedule operations if the system situation is changed, thus we apply adaptive methods for parameter adjustment.

In opposit to existing network synchronization schemes, A-MAC and ASMAC are control-center-exhaustion schemes. It is data-gathering nodes, whose energy is more abundant and easier to be recharged than data-collection nodes, that are in charge of most working load to form matching operation among nodes.

The rest of this paper is organized as follows. In Section 2, we discuss some related works. Sections 3 and 4 describe our A-MAC and ASMAC protocols, respectively. Simulation results are given in Section 5. Section 6 concludes this paper.

2. RELATED WORKS AND PRELIMINARIES

2.1. *Energy-efficient MAC protocols*

In contrast to typical MAC protocols of WLAN, MAC protocols designed for WSNs usually trade off performance (such as latency, throughput, fairness) and cost (such as energy efficiency, reduced algorithmic complexity). However, it is not clear what is the best tradeoff and various designs differ significantly.

An energy-efficient MAC protocol, power-aware multiaccess protocol with signaling (PAMAS) [10] for ad hoc networks, is proposed in 1999. PAMAS reserves battery power by intelligently powering off users that are not actively transmitting or receiving packets. In this algorithm, two separated channels—control channel and traffic channel—are needed. Following PAMAS, some other solutions for WSNs are put forward. Energy-efficient MAC protocols for WSNs can be classified into three main categories according to strategies applied to channel access: contention-based protocols, TDMA-based protocols, and slotted protocols.

As a contention-based energy-efficient MAC protocol, 802.11 [11] standard is based on carrier sensing (CSMA) and collision detection (through acknowledgements). A node intended to transmit must test the channel whether it is free for a specified time (i.e., DIFS). In [12], Hill and Culler developed a low-level carrier sensing technique that effectively turns radios off repeatedly without losing any incoming data. This technique operates at the physical layer and concerns the layout of PHY prepended header of packet. However, energy consumption by collision, overhearing, and idle listening is still an unresolved problem. Nevertheless, TDMA-based MAC protocols (i.e., TDMA) have the advantage of avoiding all those energy wastes, since TDMA scheme is inherently collision-free and schedules notify each sensor node when it should be active and, more importantly, when not.

As a TDMA-based energy-efficient MAC protocol, traffic-adaptive medium access (TRAMA) [13] employs a traffic-adaptive and distributed election scheme to allocate system time among nodes. EMACS [14] reduces idle time by forcing nodes to go into dormant mode and to wake up for announcing their presence at the schedule time only. Other TDMA-based energy-efficient MAC protocols such as bit-map-assisted (BMA) protocol and GANGS MAC protocol are described in [15, 16]. However, the price to be paid is the fixed costs (i.e., broadcasting traffic schedules) and the reduced flexibility to handle traffic fluctuations and topology change. The third type of energy-efficient MAC protocol—slotted MAC protocols—is proposed and organizes sensor nodes into a slotted system (much like slotted ALOHA), which strikes a middle ground between the first two ones.

As a slotted energy-efficient MAC protocol, S-MAC [17] is a low-power RTS-CTS protocol for WSNs inspired by PAMAS and 802.11. S-MAC includes four major components: periodic listening and sleeping, collision avoidance, overhearing avoidance, and message passing. In S-MAC, periodically listening and sleeping are designed to reduce energy consumption during the long idle time. T-MAC [18] improves S-MAC on energy usage by using a quite short listening window at the beginning of active period. To achieve ultra-low-power operation, effective collision avoidance, and high channel utilization, B-MAC [19] provides a flexible interface and employs an adaptive preamble sampling scheme to reduce duty cycle and to minimize idle listening. However, synchronization among sensor nodes is a strict premise for this kind of protocol.

Besides above works, battery-aware MAC (BAMAC(k)) protocol is proposed in [20]. BAMAC(k) is a distributed battery-aware MAC scheduling scheme, where nodes are considered as a set of batteries and scheduled by a round-robin scheduler. BAMAC(k) tries to increase the node's lifetime by exploiting the recovery capacity of batteries. Their work showed how battery awareness influences throughput, fairness, and other factors which indicate the system's performance. In [21], a power control MAC protocol, proposed power control MAC (PCM), is put forward. PCM allows nodes to vary transmission power on the packet basis, which does not degrade throughput and yields energy saving with comparison to some simple modifications of IEEE 802.11.

2.2. *Network synchronization*

For many digital communication engineers, the term synchronization is familiar in a somewhat restricted sense, meaning only the acquisition and the tracking of a clock in a receiver with reference to the periodic timing information contained in the received signal. More properly speaking, this should be referred to as carrier or symbol synchronization. Summarily, there are eight types of synchronization mainly applied to telecommunication networks, that is, carrier synchronization, symbol synchronization, frame synchronization, bit synchronization, packet synchronization, network synchronization, multimedia synchronization, and synchronization of real-time clocks [22]. Network synchronization is one of the targets in this paper.

Network synchronization deals with the distribution of time and frequency over a network spread over an even wider geographical area. The goal is to align time and frequency

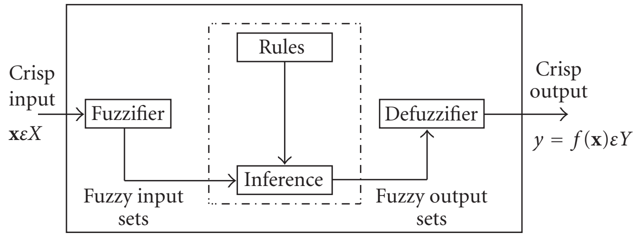


Figure 2: Structure of a fuzzy logic system.

scales of all clocks by using the communication capacity of links interconnecting them. Some well-known applications for network synchronization are synchronization of clocks located at different multiplexing and switching points in a digital telecommunication network, synchronization of clocks in a telecommunication network that requires some form of time-division multiplexing multiple access and range measurement between two nodes in a network.

Over the years, many protocols have been designed for maintaining synchronization of physical clocks over telecommunication networks [23–25]. Some wireless standards such as 802.11 have similar time-synchronization beacons built into MAC layer. Network time protocol (NTP) stands out by virtue of its scalability, self-configuration for creating a global timescale in multihop networks, robustness to various types of failure, security in the face of deliberate sabotages, and ubiquitous deployments. Other algorithms, such as time-diffusion synchronization protocol (TDP) and reference-broadcast synchronization (RBS), are proposed in [8, 26].

2.3. Preliminaries: overview of fuzzy logic systems

Figure 2 shows the structure of a fuzzy logic system (FLS) [27]. When an input is applied to an FLS, the inference engine computes the output set corresponding to each rule. The defuzzifier then computes a crisp output from these rule's output sets. Consider a p-input 1-output FLS, using singleton fuzzification, *center-of-sets* defuzzification [28], and "IF-THEN" rules of the form [29]

$$\begin{aligned} R^l:\ &\text{IF } x_1 \text{ is } F_1^l \text{ and } x_2 \text{ is } F_2^l \text{ and } \cdots \\ &\text{and } x_p \text{ is } F_p^l,\ \text{THEN } y \text{ is } G^l. \end{aligned} \tag{1}$$

Assuming singleton fuzzification, when an input $\mathbf{x}' = \{x_1', \ldots, x_p'\}$ is applied, the degree of firing corresponding to the lth rule is computed as

$$\mu_{F_1^l}(x_1') \star \mu_{F_2^l}(x_2') \star \cdots \star \mu_{F_p^l}(x_p') = \mathcal{T}_{i=1}^{p}\mu F_i^l(x_i'), \tag{2}$$

where $\star$ and $\mathcal{T}$ both indicate the chosen t-norm. There are many kinds of defuzzifiers. In this paper, we focus, for illustrative purposes, on the height defuzzifier [29]. It computes a crisp output for the FLS by first computing the height $\bar{y}^l$ of every consequent set G^l, and then computing a weighted average of these heights. The weight corresponding to the lth-rule consequent height is the degree of firing associated with the lth rule $\mathcal{T}_{i=1}^{p}\mu_{F_i^l}(x_i')$ so that

$$y_h(\mathbf{x}') = \frac{\sum_{l=1}^{M} \bar{y}^l \mathcal{T}_{i=1}^{p}\mu_{F_i^l}(x_i')}{\sum_{l=1}^{M} \mathcal{T}_{i=1}^{p}\mu_{F_i^l}(x_i')}, \tag{3}$$

where M is the number of rules in the FLS.

In [30], there is a survey on the computation complexity of the fuzzy logic system. Because a key element of fuzzy logic is its characteristic trait that transforms the binary world of digital computing into a computation based on continuous intervals, true fuzzy logic must be emulated by a software program on a standard microcontroller/processor. Inform Software Corp. has pioneered the fuzzy logic development tool market with its "fuzzyTECH microkerne" software architecture that provides implementation of fuzzy logic much more efficiently than previous emulation technologies. Now, the same example of a small fuzzy logic system running on a standard 8051 requires about one millisecond only for computation.

3. ASYNCHRONOUS MAC (A-MAC) PROTOCOL

Asynchronous MAC (A-MAC) protocol divides the system time into four phases: *TRFR-Phase, Schedule-Phase, On-Phase, and Off-Phase* (Figure 3).

(i) *TRFR-Phase* is preserved for data-collection nodes to send traffic-rate and failure-rate (TRFR) messages to data-gathering nodes.

(ii) *Schedule-Phase* is preserved for data-gathering nodes to locally broadcast phase-switching schedules.

(iii) *Off-Phase* is preserved for data-collection nodes to power off their radios. In this phase, there is no communication, but data storing and sensing may happen.

(iv) *On-Phase* is preserved for data-collection nodes to power on their radios to carry on communication.

In our system, at the end of *On-Phase*—nodes go to "vacation"—*Off-Phase*—for a period of time. Thus, new arrivals during an *On-Phase* can be served in first-in-first-out (FIFO) order. However, new arrivals during an *Off-Phase*, rather than going into service immediately, wait until the end of this *Off-Phase*, then they are served in *On-Phase* and in FIFO order. Interarrival time and service time for data packets are independent and follow general distributions $F(t)$ and $G(s)$ individually. For average interarrival time $1/\lambda$, we have $0 < 1/\lambda = \int_0^\infty \text{td}\, F(t)$. Similarly, for average service time μ, we have $0 < \mu = \int_0^\infty \text{sd}\, G(s)$.

3.1. Essential parameter design

3.1.1. Off-phase duration (T_f)

We treat each node as a single-server queuing system during our analysis on the waiting time of data packets. Note that most data packets arrive during either *Off-Phase* or *On-Phase*. The waiting time of data packet w_{ij} can be expressed

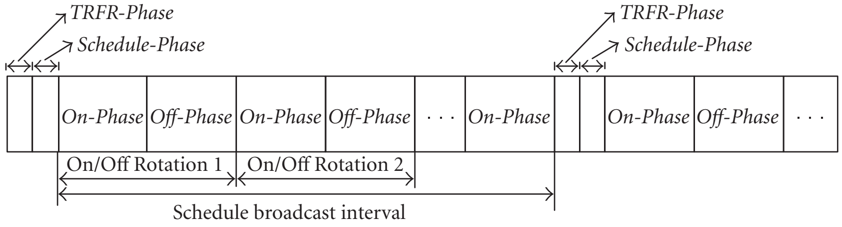


Figure 3: Time scheme structure for A-MAC.

as

$$w_{ij} = \begin{cases} T_{f,j} - \sum_{l=1}^{i} t_{lj} + \sum_{l=1}^{i-1} s_{lj} & \text{for } i = 1, 2, \dots, n, \\ \sum_{l=1}^{i-1} s_{lj} - \sum_{l=1}^{i} t_{lj} + T_{f,j} & \text{for } i = n+1, \dots, N, \end{cases} \tag{4}$$

where

(i) t_{ij} denotes the interarrival time for the ith arrived data packet for node j, and $t_{1j}, t_{2j}, \dots, t_{Nj}$ are independent and identically distributed (i.i.d.) random variables;

(ii) s_{ij} denotes the service time for the ith arrived data packet for node j, and $s_{1j}, s_{2j}, \dots, s_{Nj}$ are i.i.d. random variables also;

(iii) N is the total number of data packets that arrived during one on/off rotation, and n is the number of data packets that arrived only during an *Off-Phase*, that is, $\sum_{l=1}^{n} t_{ij} \le T_f < \sum_{l=1}^{n+1} t_{ij}$ and $\sum_{l=n+1}^{N} t_{ij} \le T_n < \sum_{l=n+1}^{N+1} t_{ij}$.

Note that w_{ij} is a function of t_{ij}, s_{ij} and $T_{f,j}$. $T_{f,j}$ is a constant for each on/off rotation during a schedule-broadcast interval. However, t_{ij} and s_{ij} are random variables with probability distribution function (PDF) $f_j(t_i) = f_j(t) = F_j'(t)$ and $g_j(s_i) = g_j(s) = G_j'(s)$, respectively. In this case, the average waiting time $\bar{w}_{ij}$ can be formulated as

$$\begin{aligned} \bar{w}_{ij} &= \int_0^\infty \cdots \int_0^\infty \int_0^\infty \cdots \int_0^\infty w_{ij} h \\ &\quad \times (t_{1j}, t_{2j}, \dots, t_{ij}, s_{1j}, s_{2j}, \dots, s_{i-1j}) dt_{ij} dt_{i-1j} \\ &\quad \cdots dt_{1j} ds_{i-1j} ds_{i-2j} \cdots ds_{1j} \\ &= \int_0^\infty \cdots \int_0^\infty \int_0^\infty \cdots \int_0^\infty w_{ij} \prod_{l=1}^{i} f_j(t_l) \prod_{l=1}^{i-1} g_j(s_l) dt_{ij} dt_{i-1j} \\ &\quad \cdots dt_{1j} ds_{i-1j} ds_{i-2j} \cdots ds_{1j}, \end{aligned} \tag{5}$$

where $h(t_{1j}, t_{2j}, \dots, t_{ij}, s_{1j}, s_{2j}, \dots, s_{i-1j})$ is the joint PDF of $t_{1j}, t_{2j}, \dots, t_{ij}$ and $s_{1j}, s_{2j}, \dots, s_{i-1j}$.

Considering λ, μ, and (4), we can rewrite (5) as follows:

$$\bar{w}_{ij} = \begin{cases} T_{f,j} - \dfrac{i}{\lambda_j} + (i-1)\mu_j & \text{for } i = 1, 2, \dots, n, \\ (i-1)\mu_j - \dfrac{i}{\lambda_j} + T_{f,j} & \text{for } i = n+1, \dots, N. \end{cases} \tag{6}$$

Note that when fixing data arrival rate λ_j and service time μ_j, the longer *Off-Phase* duration $T_{f,j}$ is, the longer data packets waiting time $\bar{w}_{ij}$ is. Moreover, based on (6), the difference of average waiting time between the ith arrived packet and the kth arrived packet for node j is shown as follows:

$$\Delta\bar{w}_j(ik) = (k-i)\left(\frac{1}{\lambda_j} - \mu_j\right) \quad (k \ge i). \tag{7}$$

Obviously, the earlier arrived data packet waits longer time than the later ones if the queuing system is not overloaded (i.e., $\mu\lambda < 1$) and is served in FIFO. In order to keep data packets up to date, $\bar{w}_{ij}$ should be no longer than the maximum acceptable waiting time $W_{\max}$, which is specified by applications. So $T_{f,j}$ should satisfy

$$T_{f,j} - \frac{1}{\lambda_j} \le W_{\max}. \tag{8}$$

T_f is the power-off duration for all nodes within a cluster. In order to ensure that data packets from all nodes are up to date, it is reasonable to choose the shortest duration of $T_{f,j}$ as a cluster's sleep duration. Then we have

$$T_f \le \min_j \left(W_{\max} + \frac{1}{\lambda_j} \right). \tag{9}$$

The expected duration, denoted by t, within which node j's buffer will be fully loaded, is given by $t = k_j/\lambda_j$, where k_j is the buffer size for node j. So $T_{f,j}$ should also satisfy the following constraint to avoid buffer overflowing:

$$T_{f,j} \le t = \frac{k_j}{\lambda_j}. \tag{10}$$

Since there are multiple nodes that have various buffer sizes and traffic arrival rates within a cluster, the power-off duration of a cluster should ensure no buffer overflow for all nodes. Hence, setting T_f equal to the shortest duration of $T_{f,j}$ determined by (10) can satisfy this criterion:

$$T_f \le \min_j \left(\frac{k_j}{\lambda_j} \right). \tag{11}$$

Combining (11) and (9), the optimum value of T_f can be obtained through

$$T_f = \min \left\{ \min_j \left(W_{\max} + \frac{1}{\lambda_j} \right), \min_j \left(\frac{k_j}{\lambda_j} \right) \right\}. \tag{12}$$

3.1.2. On-Phase duration (T_n)

During *On-Phase*, data-collection nodes start to send data packets through competition. The contention process is similar to 802.11 DCF scheme. In A-MAC algorithm, a transmission is treated as an unsuccessful one when retransmission time exceeds a threshold N_{ret}. We utilize the same model to calculate the value of N_{ret} as in [31], and only data packets will do retransmission.

If we let the duration of *On-Phase* for node j be $T_{n,j}$, according to Little's theorem [32], the total number of data packets (N_j) that arrived during an on/off rotation is given by

$$N_j = \lambda_j (T_f + T_{n,j}). \qquad (13)$$

In our *On-Phase* duration and *Off-Phase* duration designs, we not only try to extend the power-off duration to reserve energy (by avoiding excessive idle listening), but also to ensure data packets up to date. So the optimum value for $T_{n,j}$ is

$$\frac{T_{n,j}}{\mu_j} = \lambda_j (T_{n,j} + T_f). \qquad (14)$$

T_n is the power-on duration for a cluster. In order to ensure that all nodes have enough time to send buffered data packets out, we choose the longest duration of T_{nj} as a cluster's active duration:

$$T_n = \max_j \left\{ \frac{\lambda_j T_f \mu_j}{1 - \mu_j \lambda_j} \right\}. \qquad (15)$$

For 802.11 DCF scheme, the service time for data packets consists of back-off time and transmission time as follows:

$$\mu_j = \bar{T}_{B,j} + \frac{L_d}{R_j}, \qquad (16)$$

where R_j is the data transmission rate for node j and L_d is the size of the data packet.

Researches in [33, 34] showed the theoretic result on average backoff time ($\bar{T}_B$) of data transmission in 802.11 DCF scheme. Under the assumption that stations always have a packet available for transmission, in other words, the system operates in saturation condition, $\bar{T}_B$ is determined by

$$\bar{T}_B = \sum_{k=1}^{\infty} \left(\sum_{l=1}^{k} \frac{\alpha}{2} w_l \right) (1-q)^{k-1} q - \frac{\alpha}{2q} + \frac{1-q}{q} t_c, \qquad (17)$$

where

(i) q is the conditional successful probability;
(ii) $\alpha = \sigma p_i + t_c p_c + t_s p_s$;
(iii) p_s is the probability of successful transmission, p_i is the probability of the channel being idle, p_c is the probability of collision, and moreover, p_s+p_i+p_c=1;
(iv) σ is the time during which the channel is sensed idle, t_c is the average time during which the channel is sensed busy due to a collision in the channel, t_s is the average time that the common channel is sensed busy due to a successful transmission;
(v) w_l is the contention window size at the lth backoff stage.

T (2 bits)	SRC (8 bits)	AR$_\text{d}$ (8 bits)	SR$_\text{d}$ (8 bits)	FR (8 bits)	OR (8 bits)

T	Packet type	SR$_\text{d}$	Data service rate
SRC	Source address	FR	Transmission failure rate
AR$_\text{d}$	Data arrival rate	OR	Buffer overflowing rate

Figure 4: TRFR message format.

In A-MAC, for node j, the probability q_j that a packet is successfully transmitted at the end of a backoff stage is linear with its traffic strength, that is, $q_j = q\lambda_j / \sum_{l=1}^{N} \lambda_l$. We assume there are N nodes within this cluster, and each node accesses the common channel following 820.11 DCF scheme. Moreover the duration of *On-Phase* is designed to be just long enough to let all arrived data packets be sent out. In this case, the assumption for (17) is still held, but the average backoff time for our A-MAC is modified as follows:

$$\bar{T}_{B,j} = \sum_{k=1}^{\infty} \left(\sum_{l=1}^{k} \frac{\alpha}{2} w_l \right) (1-q_j)^{k-1} q_j - \frac{\alpha}{2q_j} + \frac{1-q_j}{q_j} t_c. \qquad (18)$$

3.1.3. TRFR-Phase duration

At the beginning of *TRFR-Phase*, nodes estimate their data arrival rate, service time, transmission failure rate, and buffer overflowing rate over on/off rotations independently. That information will be forwarded to data-gathering nodes through TRFR messages (Figure 4).

In this situation, data-gathering nodes become bottlenecks in increasing the chance for TRFR messages being successfully transmitted. Our strategy is to make the transmission time for each TRFR message comply with a uniform distribution, and carrier sensing is done before sending. Since hidden problem is accessible for our system, the performance will be worse compared with using CSMA/CA scheme. Following experiments shows the chance for a TRFR message being successfully transmitted.

Fixing the duration of *TRFR-Phase* from 5 to 30 seconds and increasing the number of nodes within a cluster from 5 to 30, we obtain a branch of curves on successful transmission rate for TRFR messages (Figure 5). Note that TRFR's successful transmission rate is impacted by node density (which is defined as how many nodes are there over an area) and the length of *TRFR-Phase*. From experimental results, we can choose a suitable duration for *TRFR-Phase* to ensure that data-gathering nodes can acquire necessary information from data-collection nodes to determine the system schedule successfully.

3.2. Matching schedule establishment and maintenance

According to received schedule messages (Figure 6), nodes set up their own phase-switching schedules, which ensure

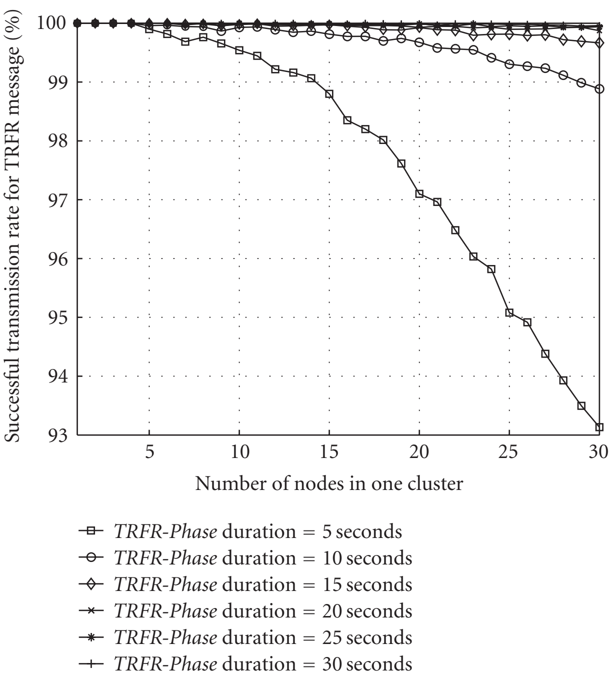


Figure 5: Successful transmission rate for TRFR message.

T (2 bits)	SRC (8 bits)	D_{TRFR} (8 bits)	D_{on} (8 bits)	D_{off} (8 bits)	I_r (8 bits)

T	Packet type	D_{on}	On-Phase duration
SRC	Source address	D_{off}	Off-Phase duration
D_{TRFR}	TRFR phase duration	I_r	Reschedule interval

Figure 6: Schedule message packet format for A-MAC.

them to switch to the same phase simultaneously. To simplify the schedule setting up process, we consider, firstly, the scenario in which there is no clock-drift and traffic is time-invariant.

We utilize two techniques to make our scheme robust and feasible to use free-running timing method [8], which allows nodes to run on their own clocks and makes contribution to save the energy used by setting up and maintaining the global or common timescale. Firstly, schedule messages are broadcasted. Leveraging the property of broadcast, schedule messages can reach all data-collection nodes at the same time, once we ignore the difference of propagation time of them (it is reasonable since the propagation time within a cluster is between 0.1 and 1 microsecond). Moreover, nodes go to *On-Phase* immediately after receiving schedule messages. Secondly, in a schedule message, all time references, such as on-duration and off-duration, are relative values rather than absolute values. This property can eliminate errors introduced by sending time and access time. Hence, each node within a cluster is synchronized to a reference packet (schedule message) that is injected into the physical channel at the same instant. Furthermore, after a same period of time specified by T_n, all nodes switch to *Off-Phase* and stay there for a T_f period. Finally, all nodes switch back to *On-Phase*. A phase is circulatedly switched like this way (see Figure 7).

Note that based on schedule messages and nodes' local clocks, phase-switching schedules are supposed to be established at each node to ensure matching operations if there is no clock-drift. Obviously, there is no global or common timescale in our system.

As we mentioned earlier, however, mismatching operations among nodes are unavoidable, since there are always clock-drifts caused by unstable and inaccurate frequency standards. So it is possible that transmitters have powered on their radios to send a message, but receivers' radios are still powered off. Those mismatching operations cause communication to fail. Moreover, with the accumulative clock-drift becoming bigger and bigger, the impact on communications turns to be more and more serious.

Our solution is to rebroadcast schedule message, which forces data-collection nodes to remove accumulative clock-drifts and to reestablish matching schedules. However, how can data-collection nodes know the time of the next schedule broadcast so as to power on their radios? The solution is that we include reschedule interval information into schedule messages. How to preestimate the value of a schedule interval is another main contribution in this paper. The details are described in Section 3.3. Flowcharts for data-gathering nodes and data-collection nodes are modified as in Figure 8, in which clock-drift is added and time-variant traffic is considered.

Nevertheless, this scheme may lose efficiency in a special situation. That is, data-collection nodes start *Schedule-Phase* later than their data-gathering nodes for accumulative clock-drifts. Consequently, the schedule broadcast will be missed and those nodes cannot be synchronized or know the latest schedule. This kind of node is named synchronization-losing node. For this issue, we design an on-demand strategy. That is, when the last *On-Phase* is over, synchronization-losing nodes proactively send requests to their data-gathering nodes. Related data-gathering nodes will reply those requests with the latest schedule and the information on the next *On-Phase*'s starting time. Then, synchronization-losing nodes can be synchronized and reestablish their phase-switching schedules.

3.3. Schedule interval design

The above discussions show that the matching operation among nodes can avoid unsuccessful transmission caused by accumulative clock-drifts. However, we also argue that it is unnecessary to offer matching operation at all times and for all nodes. For instance, two nodes, which have little information to exchange, need not to switch phases coincidently, since their mismatching operation has little influence on communications. Hence, some nodes could be allowed to go out of coincidence and to be rescheduled only if necessary.

Furthermore, from (12) and (15), we note that the durations of *On-Phase* and *Off-Phase* are tightly related to the

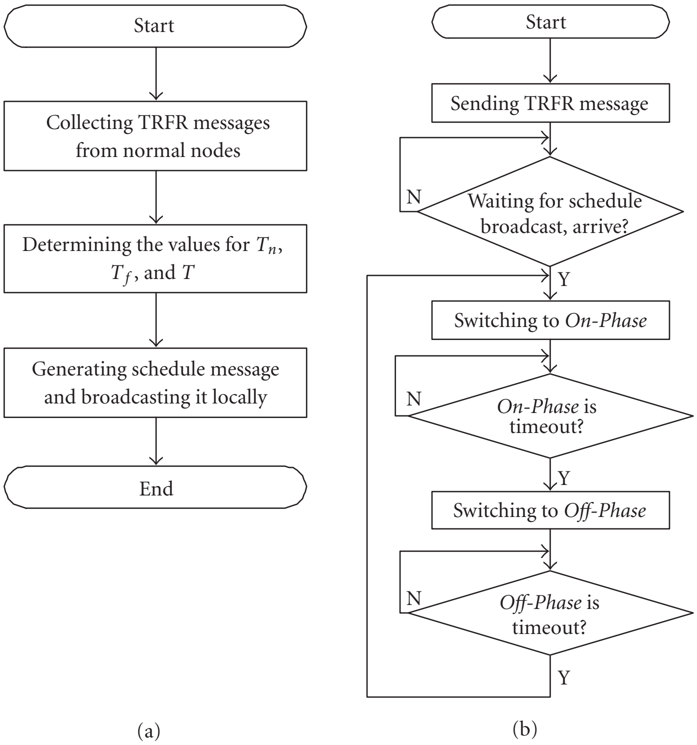

Start
Collecting TRFR messages from normal nodes
Determining the values for T_n, T_f, and T
Generating schedule message and broadcasting it locally
End
(a)
Start
Sending TRFR message
Waiting for schedule broadcast, arrive?
N
Y
Switching to *On-Phase*
On-Phase is timeout?
N
Y
Switching to *Off-Phase*
Off-Phase is timeout?
N
Y
(b)

FIGURE 7: Without clock-drift and time-variant traffic, flowchart for (a) data-gathering nodes and (b) data-collection nodes to establish and maintain matching schedules in A-MAC.

nodes' traffic strength and service capability, which are heterogeneous for WSNs as we discussed above. Thus, besides on-demandly removing accumulative clock-drifts and informing phase-switching schedules, an additional function for schedule broadcasts is to acquire more suitable values for essential parameters according to the system situation. This property enables our algorithm to be an adaptive scheme in terms of node density and traffic strength.

How to combine all factors to adjust the length of schedule interval correctly is a complicated and vague task, which impacts the performance in terms of energy reservation and successful communication significantly. Since FLS is outstanding in dealing with uncertain problems, we design a rescheduling FLS to monitor the influence of accumulative clock-drifts, the variance of traffic strength, and service capability on communications. Then we can adjust schedule interval and power-on/off duration adaptively. We use

$$T_i = \xi_i \times T_{i-1} \tag{19}$$

as our interval adjustment function, where T_i is the interval for the ith schedule broadcast, ξ_i is the ith adjustment factor determined by our rescheduling FLS.

In our rescheduling FLS, there are three antecedents:

(i) ratio of node with overflowing buffer (R_{of}): the percentage of node having buffer overflowing within a cluster;

(ii) ratio of node with high unsuccessful transmission rate (R_{hf}): the percentage of node whose unsuccessful transmission rate is higher than a threshold within a cluster;

(iii) ratio of node experiencing unsuccessful transmission (R_{sr}): the percentage of node having transmission failure within a cluster.

R_{of} reflects traffic strength. R_{hf} and R_{sr} reflect the influence of accumulative clock-drifts on communications from depth and width aspects individually.

The consequent is the adjustment factor (ξ_i) for the schedule-broadcast interval. The linguistic variables representing R_{of}, R_{hf}, and R_{sr} are divided into three levels: *low, moderate*, and *high*. ξ_i is divided into 5 levels: *highly decrease, decrease, unchange, increase*, and *highly increase*. We use trapezoidal membership functions (MFs) to represent *low, high, highly decrease*, and *highly increase*, and triangle MFs to represent *moderate, decrease, unchange*, and *increase*. We show those MFs in Figures 9(a) and 9(b).

The schedule interval should be shortened when there are many data packets missing due to accumulative clock-drifts and/or unsuitable *Off-Phase* duration, otherwise the schedule interval should be extended to reduce the energy consumption on scheduling. Based on this fact, we design our rescheduling FLS using rules summarized in Table 1.

For every input (R_{of}, R_{hf}, R_{sr}), the output is defuzzified using (20). The heights of the five fuzzy sets depicted in

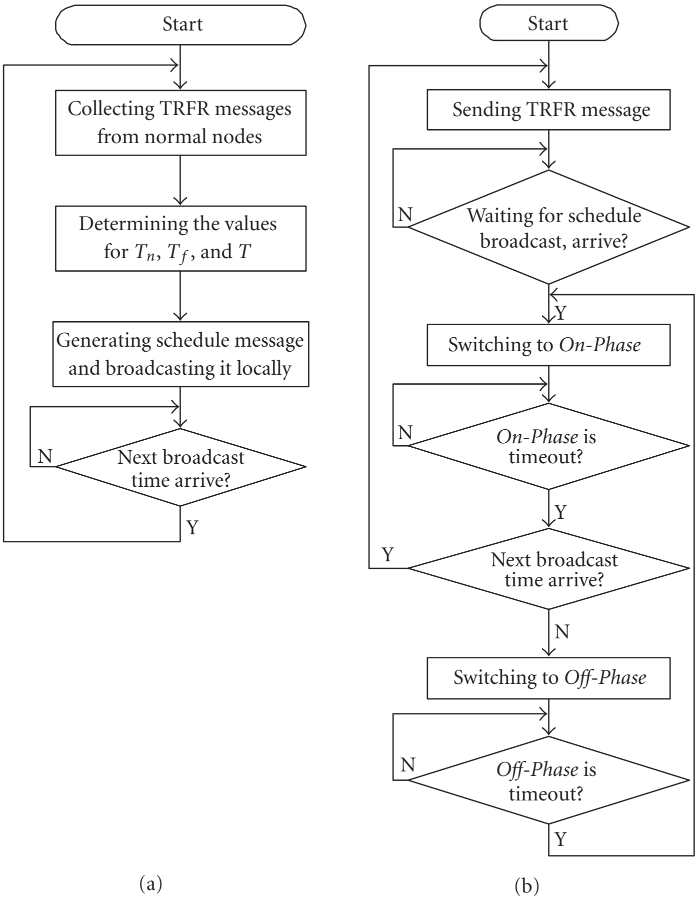

Start
Collecting TRFR messages from normal nodes
Determining the values for T_n, T_f, and T
Generating schedule message and broadcasting it locally
Next broadcast time arrive?
N
Y
(a)
Start
Sending TRFR message
Waiting for schedule broadcast, arrive?
N
Y
Switching to *On-Phase*
On-Phase is timeout?
N
Y
Next broadcast time arrive?
Y
N
Switching to *Off-Phase*
Off-Phase is timeout?
N
Y
(b)

Figure 8: With clock-drift and time-variant traffic, flowchart for (a) data-gathering nodes and (b) data-collection nodes to establish and maintain matching schedules in A-MAC.

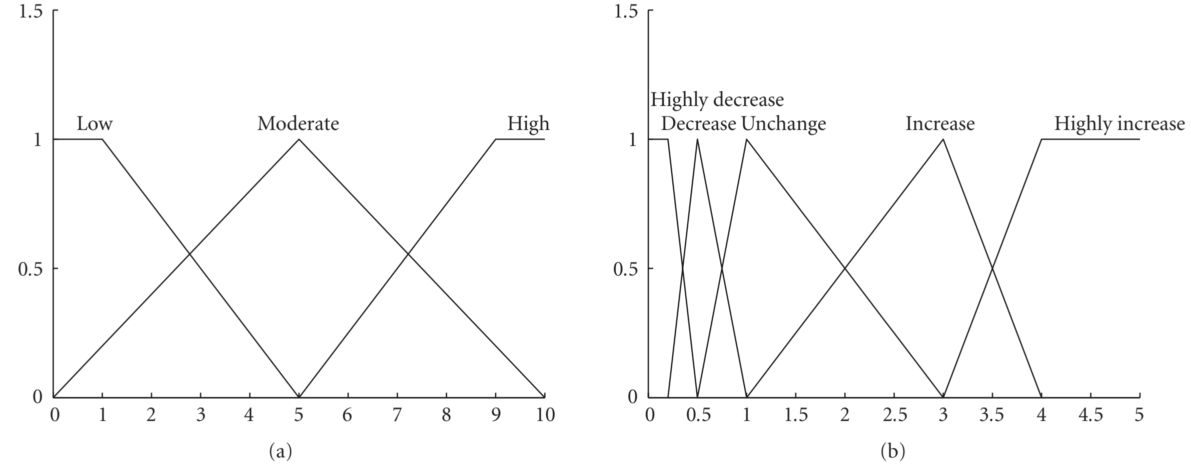


Figure 9: (a) Antecedent MFs for rescheduling-FLS and (b) consequent MFs for rescheduling-FLS.

Figure 9(b) are $\bar{\xi}_1 = 0.2$, $\bar{\xi}_2 = 0.5$, $\bar{\xi}_3 = 1.0$, $\bar{\xi}_4 = 3.0$, $\bar{\xi}_5 = 4.0$,

$$y(R_{\text{of}}, R_{\text{hf}}, R_{\text{sr}}) = \frac{\sum_{l=1}^{15} \bar{\xi}^l \mu_{F_1^l}(R_{\text{of}})\mu_{F_2^l}(R_{\text{hf}})\mu_{F_3^l}(R_{\text{sr}})}{\sum_{l=1}^{15} \mu_{F_1^l}(R_{\text{of}})\mu_{F_2^l}(R_{\text{hf}})\mu_{F_3^l}(R_{\text{sr}})}. \quad (20)$$

The inputs of rescheduling FLS are acquired from TRFR messages. Prior to each schedule broadcast, rescheduling FLSs located in data-gathering nodes individually estimate the influence degree of the accumulative clock-drift and the change of traffic strength on communications. After

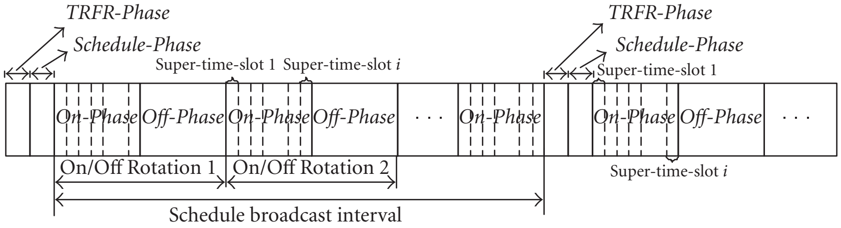


FIGURE 10: System time scheme structure for ASMAC.

TABLE 1: Rules for rescheduling-FLS; Ante1 is R_{of}, Ante2 is R_{hf}, Ante3 is R_{sr}, consequent is ξ.

Rule	Ante1	Ante2	Ante3	Consequent
1	Low	Low	Low	Highly increase
2	Low	Low	Moderate	Increase
3	Low	Moderate	Moderate	Decrease
4	Low	Moderate	High	Decrease
5	Moderate	Low	Moderate	Increase
6	Moderate	Low	High	Unchange
7	Moderate	Moderate	Moderate	Decrease
8	Moderate	Moderate	High	Highly decrease
9	Low	High	High	Decrease
10	Moderate	High	High	Highly decrease
11	High	Low	Moderate	Increase
12	High	Low	High	Unchange
13	High	Moderate	Moderate	Decrease
14	High	Moderate	High	Decrease
15	High	High	High	Highly decrease

obtaining ξ_i, data-gathering nodes determine the value for the next schedule-broadcast interval according to (19). This operation cannot only save energy through avoiding unnecessary schedule broadcasts and idle listening, but also ensures an adequate data successful transmission rate.

4. ASYNCHRONOUS SCHEDULE-BASED MAC (ASMAC) PROTOCOL

Asynchronous schedule-based MAC (ASMAC) is similar to A-MAC. ASMAC's system time is also divided into four phases: *TRFR-Phase, Schedule-Phase, On-Phase*, and *Off-Phase* (Figure 10). The same TRFR message and *TRFR-Phase* duration design method are used by ASMAC. However, *On-Phase* is further divided into super-time-slots, which are composed of several normal time slots, and one source-destination pair continuously occupies one super-time-slot.

T (2 bits)	SRC (8 bits)	D_{off} (8 bits)	D_{on} (8 bits)
SRC_1 (8 bits)	DEST_1 (8 bits)	D_{df1} (8 bits)	D_{s1} (8 bits)
SRC_2 (8 bits)	DEST_2 (8 bits)	D_{df2} (8 bits)	D_{s2} (8 bits)
– –			
SRC_i (8 bits)	DEST_i (8 bits)	$\text{D}_{\text{df}i}$ (8 bits)	$\text{D}_{\text{s}i}$ (8 bits)

T	Packet type
SRC	Source address
D_{off}	Off-Phase duration
D_{on}	On-Phase duration
D_{s1}	super-time-slot duration for node *i*
$\text{D}_{\text{df}i}$	super-time-slot starts defer time
SRC_1	Source address for *i*th super-time-slot
DEST_i	Destination address for *i*th super-time-slot

FIGURE 11: Schedule message packet format for ASMAC.

We add ACK message as the acknowledgment for receiving data packets successfully. A transmission is defined as an unsuccessful one once the transmitter does not receive ACK after a certain period of time. The format of the schedule message is shown in Figure 11.

Matching schedule establishment, maintenance, and schedule interval design mechanisms of ASMAC are the same as in A-MAC, but power-on/off duration design is somewhat different. A new task, time-slot allocation, is added into ASMAC.

4.1. On-Phase/Off-Phase duration (T_n/T_f) design

In ASMAC, nodes perform communication in their own super-time-slots and turn off their radios to save energy in *Off-Phase* and other nodes' super-time-slots. Hence, nodes carry out communications orderly and contention freely. The same criteria are utilized by ASMAC for *On-Phase* and *Off-Phase* duration designs: trying to save more energy, keeping information up to date, and avoiding losing information due to buffer overflowing. The optimum values for T_f and

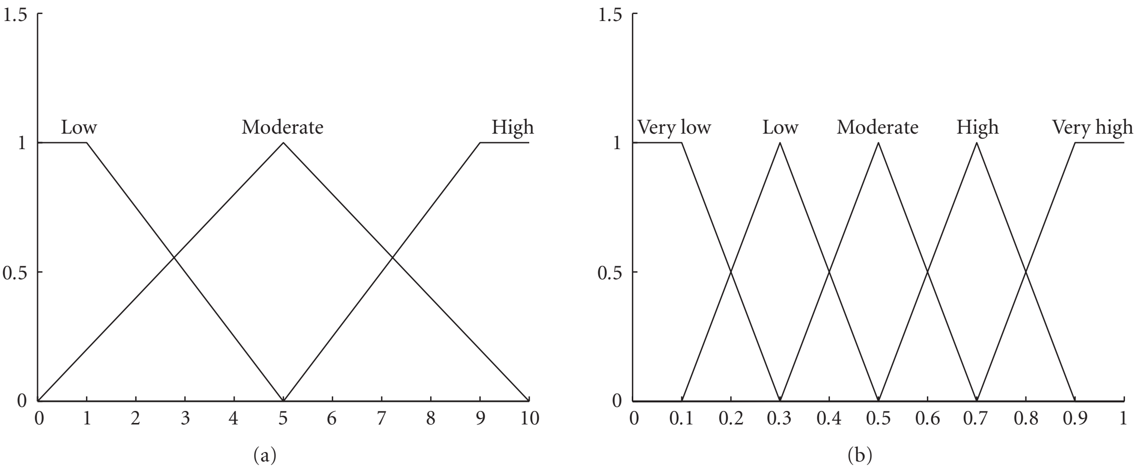


FIGURE 12: (a) Antecedent MFs for allocation-FLS and (b) consequent MFs for allocation-FLS.

T_n can be calculated:

$$
\begin{aligned}
&(1)\ \text{when } 2W_{\max} < \min_j \left(\frac{k_j}{\lambda_j}\right),\\
&\quad T_f = \frac{2W_{\max}}{\left(1+\sum_{l=1}^{N}\left(\mu_l\lambda_l/(1-\mu_l\lambda_l)\right)\right)},\\
&\quad T_n = \frac{2W_{max}\sum_{l=1}^{N}\left(\mu_l\lambda_l/(1-\mu_l\lambda_l)\right)}{\left(1+\sum_{l=1}^{N}\left(\mu_l\lambda_l/(1-\mu_l\lambda_l)\right)\right)};\\
&(2)\ \text{when } 2W_{\max} \geq \min_j \left(\frac{k_j}{\lambda_j}\right),\\
&\quad T_f = \min_j\left(\frac{k_j/\lambda_j}{1+\sum_{l=1}^{N}\left(\mu_l\lambda_l/(1-\mu_l\lambda_l)\right)}\right),\\
&\quad T_n = \min_j\left(\frac{\left(k_j/\lambda_j\right)\sum_{l=1}^{N}\left(\mu_l\lambda_l/(1-\mu_l\lambda_l)\right)}{1+\sum_{l=1}^{N}\left(\mu_l\lambda_l/(1-\mu_l\lambda_l)\right)}\right).
\end{aligned}
\tag{21}
$$

4.2. *Time-slot assignment*

For classic TDMA systems, the system time is divided into slots, and each user occupies cyclically repeating time slots—a buffer-and-burst method. Thus, high-quality network synchronization method is needed. Unfortunately, this premise is troublesome for our ASMAC scheme. However, we note that as the length of time slots increases, more transmissions are done successfully under the same mismatching situation. Hence, with contrast to buffer-and-burst method, we design a buffer-and-continue method to enhance the tolerance on accumulative clock-drifts, in which the same communication pairs occupy sets of continuous time-slots.

In ASMAC, we design an allocation FLS to correspondingly quantify transmission priorities for each node. There are two antecedents to our allocation FLS:

(i) traffic arrival rate (R_a),
(ii) transmission failure rate (R_{us}).

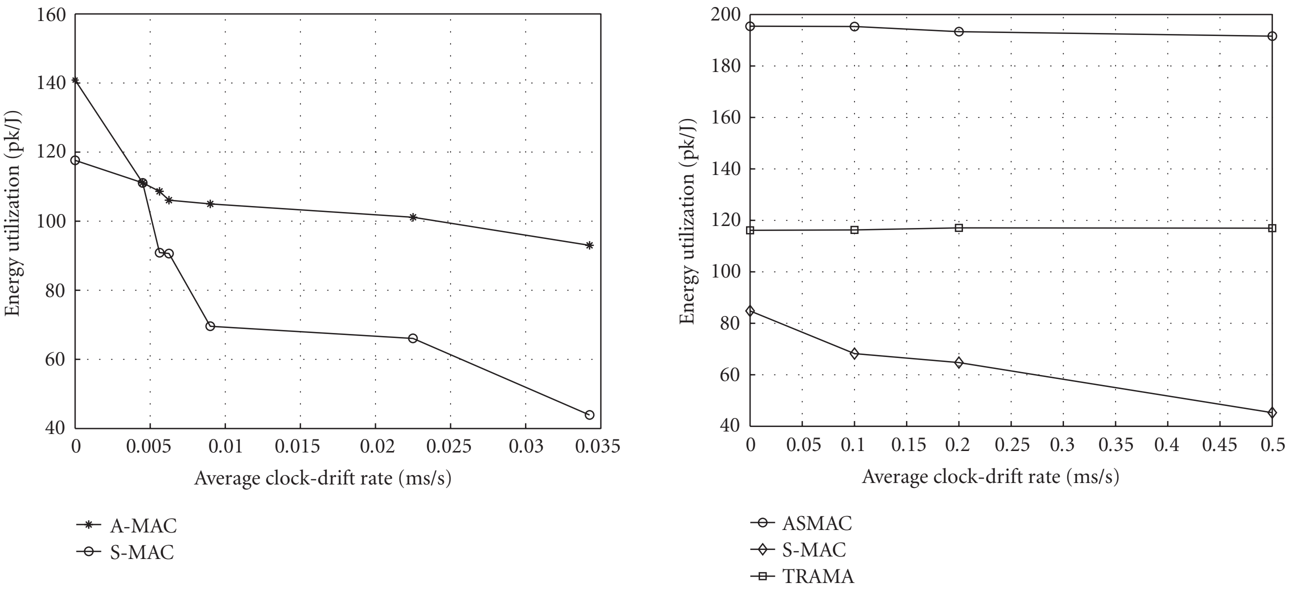
TABLE 2: Rules for allocation-FLS; Ante1 is the traffic arrival rate, Ante2 is the unsuccessful transmission rate, consequent is the priority of a node performing transmission.

Rule	Ante1	Ante2	Consequent
1	Low	Low	Moderate
2	Low	Moderate	High
3	Low	High	Very high
4	Moderate	Low	Low
5	Moderate	Moderate	Moderate
6	Moderate	High	High
7	High	Low	Very low
8	High	Moderate	Low
9	High	High	Moderate

The consequent is the priority of a node performing transmission (P_t).

The linguistic variables used to represent R_a and R_{us} are divided into three levels: *low, moderate*, and *high*. P_t is divided into 5 levels: *very hign, high, moderate, low*, and *very low*. We use trapezoidal membership functions (MFs) to represent *low, high, very low*, and *very high*, and triangle MFs to represent *moderate, low*, and *high*. We show those MFs in Figures 12(a) and 12(b).

The transmission priority of a node should be higher when there are more data packets waiting for transmitting and/or its transmission failure rate is high. Based on this fact, we design our allocation FLS using rules summarized in Table 2. With the allocation FLS, data-gathering nodes leverage the information acquired from TRFR messages to quantify priorities for nodes. The node owning the highest priority is the earliest one to perform communications during *On-Phase*.



(a)

(b)

Figure 13: Energy utilization for (a) A-MAC and (b) ASMAC.

Table 3: Physical layer parameters.

$W_{w\min}$	32	$W_{w\max}$	1024
MAC header	34 bytes	ACK	38 bytes
CTS	38 bytes	RTS	44 bytes
SIFS	10 μs	DIFS	50 μs
ACK timeout	212 μs	CTS timeout	348 μs

5. SIMULATION AND PERFORMANCE EVALUATION

We used the simulator OPNET to run simulations. A network with 30 nodes is set up and the radio range (radius) of each node is 30 m. Those nodes are randomly deployed in an area of $100 \times 100\,\text{m}^2$ and have no mobility. This network can be treated as one cluster in a large-scale system. In order to simplify the analysis about the impact of accumulative clock-drifts on communications and the performance of our MAC algorithms, we exclude the factors coming from physical layer and network layer in our experiments. The clock-drift rate of frequency oscillators varies from 1 to 100 microseconds every second. Table 3 summarizes the parameters used by our simulations. The packet size is 1000 bytes. The destination for each node's traffic is randomly chosen from its neighbors.

As in [35, 36], data packets arrive according to a Poisson process with certain rate in our simulations. Moreover, every 10 seconds, the traffic will be held for 5 seconds to simulate bursting traffic. In our simulations, we substitute statistic average values with time-average values for data packet arrival rate and service time.

All nodes are set with initial energy of 15 J. We use the same energy consumption model as in [37] for radio hardware. To transmit an l-symbol message for a distance d, the radio expends:

$$\begin{aligned} E_{\text{Tx}}(l,d) &= E_{\text{Tx}-\text{elec}(l)} + T_{\text{Tx}-\text{amp}(l,d)} \\ &= l \times E_{\text{elec}} + l \times e_{\text{fs}} \times d^2; \end{aligned} \tag{22}$$

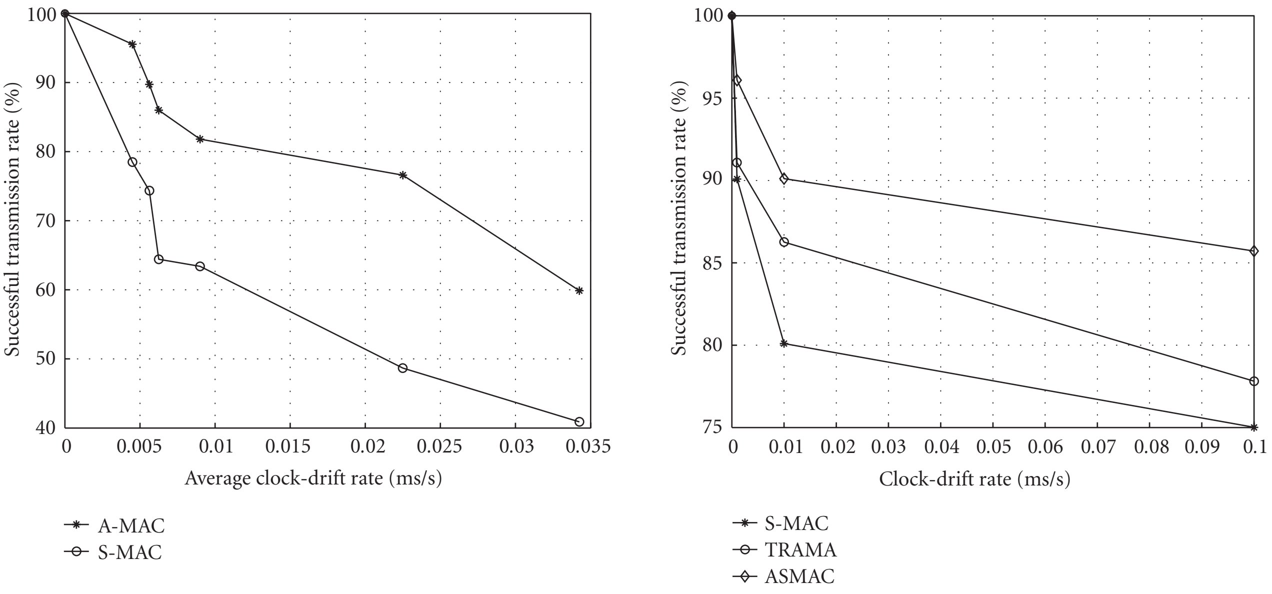
and to receive this message, the radio expends:

$$E_{\text{Rx}} = l \times E_{\text{elec}}. \tag{23}$$

The electronics energy E_{elec}, as described in [37], depends on factors such as coding, modulation, pulse shaping, and matched filtering. The amplifier energy $e_{\text{fs}} \times d^2$ depends on the distance to the receiver and the acceptable bit error rate. In this paper, we choose $E_{\text{elec}} = 50\,\text{nJ/syn}$ and $e_{\text{fs}} = 10\,\text{pJ/sym/m}^2$. When a node receives packets but the destination is not for it, those packets will be discarded. This kind of useless receiving, that is, idle listening, uses the same model in (23) to calculate energy consumption.

5.1. *A-MAC versus S-MAC*

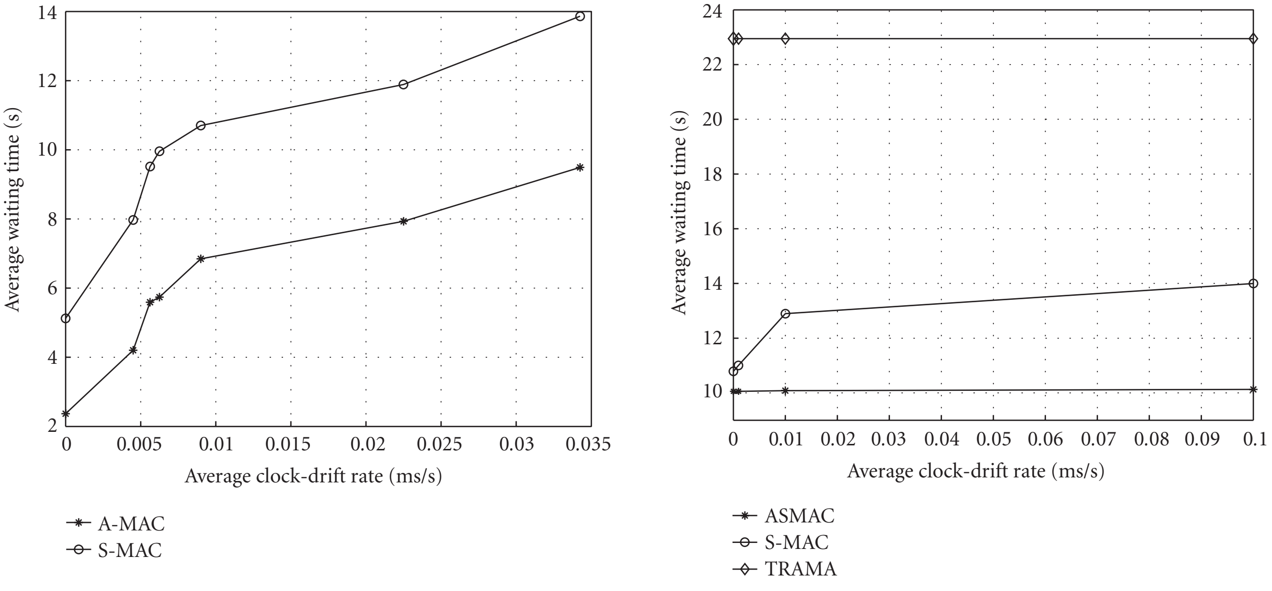
We compared our A-MAC against S-MAC [17] without network synchronization function. In our simulations, energy utilization is assessed by the number of successfully transmitted data packets per J, and the unit is pk/J. Energy utilization versus clock-drift rate is plotted in Figure 13(a). Observe that A-MAC can send 17.85% to 33.33% more packets per J. Therefore, with the same available energy and traffic strength, the lifetime of a network will be extended about 0.2 to 0.4 times when using our algorithm A-MAC instead of S-MAC scheme. This result demonstrates that A-MAC can implement the energy reservation task successfully.



(a)

(b)

Figure 14: Successful transmission rate for (a) A-MAC and (b) ASMAC.



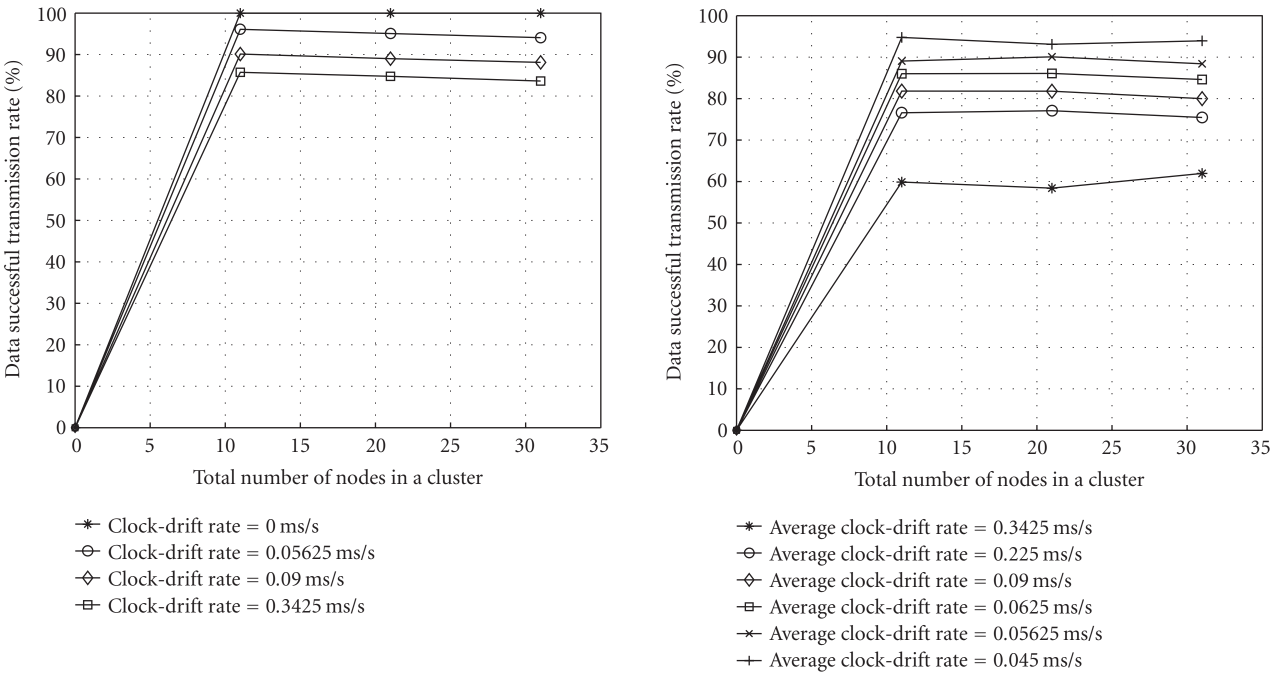
(a)

(b)

Figure 15: Waiting time for (a) A-MAC and (b) ASMAC.

Saving energy is one of the main aims of A-MAC. However, we should not achieve this goal through sacrificing data successful transmission rate, since communication is the ultimate role for communication systems. In Figure 14(a), we plot data successful transmission rate versus clock-drift rate. Observe that A-MAC can transmit data packets successfully about 30.77% more than S-MAC. The reasons are, firstly, failure transmissions are reduced because clock-drifts among nodes are removed effectively; secondly, more energy is utilized by transmitting data packets.

In Figure 15(a), we plot average waiting time versus clock-drift rate. It is shown that A-MAC has about 33.3% shorter waiting time than that of S-MAC. Moreover, we set $W_{\max}$ to 12 seconds for average waiting time, we found that average waiting-time for A-MAC is always shorter than 12 seconds even at different clock-drift rates. However for



(a)

(b)

Figure 16: Network density adaptation for (a) ASMAC and (b) A-MAC.

S-MAC, the average waiting-time is longer than 12 seconds when clock-drift rate is longer than 0.0225 ms/s. That is, there are many out-of-date packets received when using S-MAC. This result demonstrates our claim that our algorithm A-MAC is a waiting-time-aware method.

5.2. ASMAC versus S-MAC and TRAMA

We compared our ASMAC against S-MAC and TRAMA [13] without network synchronization function. Energy utilization versus clock-drift rate is plotted in Figure 13(b). Observe that ASMAC can send 41.176% to 56.14% more packets per J. Therefore, with same available energy and traffic strength, the lifetime of a network will be extended about 0.4 to 0.6 times when using our algorithm ASMAC instead of S-MAC and TRAMA schemes. This result demonstrates that ASMAC can also implement energy reservation task successfully.

In Figure 14(b), we plot data successful transmission rate versus clock-drift rate. Observe that ASMAC can transmit data packets successfully about 12.5% more than S-MAC, and about 4.65% more than TRAMA.

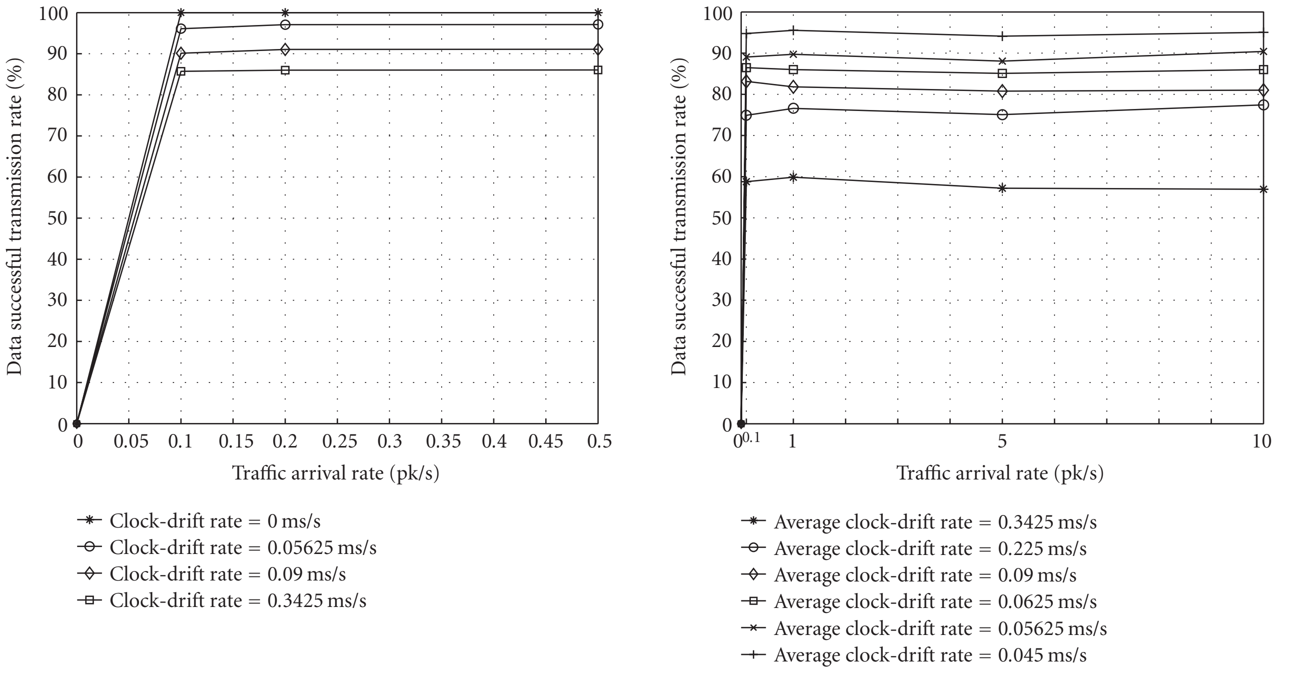
In Figure 15(b), we plot average waiting time versus clock-drift rate. It is shown that ASMAC has about 56.178% shorter waiting time than TRAMA, and about 8.648% than S-MAC. We found that the average waiting time for ASMAC is also shorter than $W_{\max} = 12$ seconds at different clock-drift rates. However, for TRAMA and S-MAC, the average waiting time is longer than that threshold.

5.3. Adaptation of ASMAC and A-MAC

We investigate the influences of node density and traffic strength on system performance of our algorithms. We change node density and traffic strength individually at a set of clock-drift situations. In Figure 16(a), we plot number of nodes, changed from 10 to 30, in a cluster versus successful transmission rate of data packet for ASMAC. Notice that for each clock-drift rate, the vibration of successful transmission rate with the change of the node density is less than 85.714% − 83.606% = 2.108%. The same experiment is done for A-MAC. We can see that the vibration of successful transmission rate is less than 61.96% − 59.87% = 2.09% (Figure 16(b)).

In Figure 17(a), we compare successful transmission rate at different traffic arrival rates, varying from 0.1 to 0.5 pk/s. This shows that for each clock-drift, the vibration of successful transmission rate with the change of node number is less than 97.099% − 96.087% = 1.012% for ASMAC. The vibration of A-MAC is less than 60% − 58.79% = 1.21% (Figure 17(b)).

These two experiments show that with the variance of node density and traffic strength, network throughput can keep almost stable through using our A-MAC and AS-MAC protocols. The reason is that essential parameters—reschedule interval, *On-Phase*, and *Off-Phase* durations—are adaptively adjusted with the system situation.



(a)

(b)

Figure 17: Traffic intensity adaptation for (a) ASMAC and (b) A-MAC.

6. CONCLUSIONS

In this paper, we proposed two energy-efficient MAC protocols for WSNs: A-MAC and ASMAC. They make following contributions compared with existing energy-efficient MAC protocols for WSNs:

(i) saving energy at MAC layer through trading off data waiting time and reducing energy consumption on collision and idle listening;

(ii) utilizing free-running time scheme and schedule broadcast to set up system schedules without establishing a common timescale within a system;

(iii) exploiting a reschedule method, instead of network synchronization, to handle mismatching operations caused by accumulative clock-drifts;

(iv) taking advantage of fuzzy logical theories to design rescheduling FLS and allocation FLS;

(v) proposing a traffic-strength- and network-density-based model to optimize essential algorithm parameters.

Simulation results showed that not only the performance of network is improved, but also its lifetime is extended when A-MAC or ASMAC is used.

ACKNOWLEDGMENT

This work was supported by the US Office of Naval Research (ONR) Young Investigator Program Award under Grant N00014-03-1-0466.

REFERENCES

[1] D. Culler, D. Estrin, and M. Srivastava, "Guest Editors' Introduction: overview of sensor networks," *Computer*, vol. 37, no. 8, pp. 41–49, 2004.

[2] F. Zhao and L. Guibas, *Wireless Sensor Networks: An Information Processing Approach*, Morgan Kaufmann, San Francisco, Calif, USA, 2004.

[3] M. Stemm and R. H. Katz, "Measuring and reducing energy consumption of network modules in hand-held devices," *IEICE Transactions on Communications*, vol. E80-B, no. 8, pp. 1125–1131, 1997.

[4] J. Chou, D. Petrovic, and K. Ramachandran, "A distributed and adaptive signal processing approach to reducing energy consumption in sensor networks," in *Proceedings of 22nd Annual Joint Conference on the IEEE Computer and Communications Societies (INFOCOM '03)*, vol. 2, pp. 1054–1062, San Francisco, Calif, USA, March-April 2003.

[5] M. L. Chebolu, V. K. Veeramachaneni, S. K. Jayaweera, and K. R. Namuduri, "An improved adaptive signal processing approach to reduce energy consumption in sensor networks," in *Proceedings of 38th Annual Conference on Information Science and System (CISS '04)*, Princeton, NJ, USA, March 2004.

[6] S. Balasubramanian, I. Elangovan, S. K. Jayaweera, and K. R. Namuduri, "Distributed and collaborative tracking for energy-constrained ad-hoc wireless sensor networks," in *Proceedings of IEEE Wireless Communications and Networking Conference (WCNC '04)*, vol. 3, pp. 1732–1737, Atlanta, Ga, USA, March 2004.

[7] S. K. Jayaweera, "An energy-efficient virtual MIMO communications architecture based on V-BLAST processing for distributed wireless sensor networks," in *Proceedings of 1st Annual IEEE Communications Society Conference on Sensor and*

Ad Hoc Communications and Networks (SECON '04), pp. 299–308, Santa Clara, Calif, USA, October 2004.

[8] J. E. Elson, "Time synchronization in wireless sensor networks," Dissertation, Computer Science Department, University of California Los Angeles, Los Angeles, Calif, USA, 2003.

[9] C. Ma, M. Ma, and Y. Yang, "Data-centric energy efficient scheduling for densely deployed sensor networks," in *Proceedings of IEEE International Conference on Communications*, vol. 6, pp. 3652–3656, Paris, France, June 2004.

[10] S. Singh and C. S. Raghavendra, "Pamas: power aware multiaccess protocol with signaling for ad hoc networks," *ACM SIGCOMM Computer Communication Review*, vol. 28, no. 3, pp. 5–26, 1998.

[11] P802.11, "Ieee standard for wireless lan medium access control (mac) and physical layer (phy) specifications," November 1997.

[12] J. L. Hill and D. E. Culler, "Mica: a wireless platform for deeply embedded networks," *IEEE Micro*, vol. 22, no. 6, pp. 12–24, 2002.

[13] V. Rajendran, K. Obraczka, and J. J. Garcia-Luna-Aceves, "Energy-efficient, collision-free medium access control for wireless sensor networks," in *Proceedings of the 1st International Conference on Embedded Networked Sensor Systems (SenSys '03)*, pp. 181–192, Los Angeles, Calif, USA, November 2003.

[14] L. F. W. van Hoesel, T. Nieberg, H. J. Kip, and P. J. M. Havinga, "Advantages of a TDMA based, energy-efficient, self-organizing MAC protocol for WSNs," in *Proceedings of IEEE 59th Vehicular Technology Conference (VTC '04)*, vol. 3, pp. 1598–1602, Milan, Italy, May 2004.

[15] J. Li and G. Y. Lazarou, "A bit-map-assisted energy-efficient MAC scheme for wireless sensor networks," in *Proceedings of 3rd International Symposium on Information Processing in Sensor Networks (IPSN '04)*, pp. 55–60, Berkeley, Calif, USA, April 2004.

[16] S. Biaz and Y. D. Barowski, "GANGS: an energy efficient MAC protocol for sensor networks," in *Proceedings of the 42nd Annual Southeast Regional Conference (ACMSE '04)*, pp. 82–87, Huntsville, Ala, USA, April 2004.

[17] W. Ye, J. Heidemann, and D. Estrin, "An energy-efficient MAC protocol for wireless sensor networks," in *Proceedings of 21st Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '02)*, vol. 3, pp. 1567–1576, New York, NY, USA, June 2002.

[18] T. Van Dam and K. Langendoen, "An adaptive energy-efficient MAC protocol for wireless sensor networks," in *Proceedings of the 1st International Conference on Embedded Networked Sensor Systems (SenSys '03)*, pp. 171–180, Los Angeles, Calif, USA, November 2003.

[19] J. Polastre, J. Hill, and D. Culler, "Versatile low power media access for wireless sensor networks," in *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems (SenSys '04)*, pp. 95–107, Baltimore, Md, USA, November 2004.

[20] S. Jayashree, B. S. Manoj, and C. S. R. Murthy, "On using battery state for medium access control in ad hoc wireless networks," in *Proceedings of the 10th Annual International Conference on Mobile Computing and Networking (MobiCom '04)*, pp. 360–373, Philadelphia, Pa, USA, September-October 2004.

[21] E.-S. Jung and N. H. Vaidya, "A power control MAC protocol for ad hoc networks," in *Proceedings of the 8th Annual International Conference on Mobile Computing and Networking (MobiCom '02)*, pp. 36–47, Atlanta, Ga, USA, September 2002.

[22] S. Bregni, *Synchronization of Digital Telecommunications Networks* , John Wiley & Sons, New York, NY, USA, 2002.

[23] F. Cristian, "Probabilistic clock synchronization," *Distributed Computing*, vol. 3, no. 3, pp. 146–158, 1989.

[24] R. Gusella and S. Zatti, "The accuracy of the clock synchronization achieved by TEMPO in Berkeley UNIX 4.3 BSD," *IEEE Transactions on Software Engineering*, vol. 15, no. 7, pp. 847–853, 1989.

[25] T. K. Srikanth and S. K. Toueg, "Optimal clock synchronization," *Journal of the ACM*, vol. 34, no. 3, pp. 626–645, 1987.

[26] W. Su and I. F. Akyildiz, "Time-diffusion synchronization protocol for wireless sensor networks," *IEEE/ACM Transactions on Networking*, vol. 13, no. 2, pp. 384–397, 2005.

[27] J. M. Mendel, "Fuzzy logic systems for engineering: a tutorial," *Proceedings of the IEEE*, vol. 83, no. 3, pp. 345–377, 1995.

[28] E. H. Mamdani, "Application of fuzzy logic to approximate reasoning using linguistic synthesis," *IEEE Transactions on Computers*, vol. 26, no. 12, pp. 1182–1191, 1977.

[29] J. M. Mendel, *Uncertain Rule-Based Fuzzy Logic Systems: Introduction and New Directions*, Prentice-Hall, Upper Saddle River, NJ, USA, 2001.

[30] C. V. Altrock, "Fuzzy logic design: methodology, standards, and tools," *Electronic Engineering Times*, July 1996.

[31] L. H. Bao and J. J. Garcia-Luna-Aceves, "Hybrid channel access scheduling in ad hoc networks," in *Proceedings of 10th IEEE International Conference on Network Protocols (ICNP '02)*, pp. 46–57, Paris, France, November 2002.

[32] D. Bertsekas and R. Gallager, *Data Networks*, Prentice-Hall, Upper Saddle River, NJ, USA, 1987.

[33] M. M. Carvalho and J. J. Garcia-Luna-Aceves, "Delay analysis of IEEE 802.11 in single-hop networks," in *Proceedings of 11th IEEE International Conference on Network Protocols (ICNP '03)*, pp. 146–155, Atlanta, Ga, USA, November 2003.

[34] G. Bianchi, "Performance analysis of the IEEE 802.11 distributed coordination function," *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 3, pp. 535–547, 2000.

[35] A. Manjeshwar, Q.-A. Zeng, and D. P. Agrawal, "An analytical model for information retrieval in wireless sensor networks using enhanced APTEEN protocol," *IEEE Transactions on Parallel and Distributed Systems*, vol. 13, no. 12, pp. 1290–1302, 2002.

[36] V. P. Mhatre, C. Rosenberg, D. Kofman, R. Mazumdar, and N. Shroff, "A minimum cost heterogeneous sensor network with a lifetime constraint," *IEEE Transactions on Mobile Computing*, vol. 4, no. 1, pp. 4–14, 2005.

[37] W. B. Heinzelman, A. P. Chandrakasan, and H. Balakrishnan, "An application-specific protocol architecture for wireless microsensor networks," *IEEE Transactions on Wireless Communications*, vol. 1, no. 4, pp. 660–670, 2002.

Qingchun Ren received her B.S. and M.S. degrees from University of Electrical Science and Technology of China, in 1997 and 2003, respectively, both in electrical engineering. She is working towards the Ph.D. degree in electrical engineering at The University of Texas at Arlington. Prior to that, she was a Member of the technical staff at WATT Electronic Co., Ltd. at Shenzhen, China. Since August 2003, she has been a Research Assistant in the Wireless Communication Network Group, The University of Texas at Arlington. Her research interests are in sensor networks (energy efficiency, cross-layer design, optimal sensor deployment, etc.), fuzzy logic systems, and query processing for sensor database systems.

Qilian Liang received the B.S. degree from Wuhan University, China, in 1993, the M.S. degree from Beijing University of Posts and Telecommunications in 1996, and the Ph.D. degree from University of Southern California (USC) in May 2000, all in electrical engineering. He joined the faculty of The University of Texas at Arlington in August 2002. Prior to that, he was a Member of the technical staff in Hughes Network Systems Inc. at San Diego, California. His research interests include sensor networks (energy efficiency, cross-layer design, optimal sensor deployment, etc.), wireless communications, wireless networks, communication theory, signal processing for communications, fuzzy logic systems and applications, multimedia network traffic modeling and classification, collaborative and distributed signal processing. He has published more than 90 journal and conference papers, 4 book chapters, and has 6 US patents pending. He received 2002 IEEE Transactions on Fuzzy Systems Outstanding Paper Award, 2003 US Office of Naval Research (ONR) Young Investigator Award, and 2005 UTA College of Engineering Outstanding Young Faculty Award.