Quorum-based Localized Scheme for Duty Cycling in Asynchronous Sensor Networks

Jianhui Zhang*‡, Shao-jie Tang†, Xingfa Shen*, Guojun Dai*, Amiya Nayak‡

*College of Computer Science and Technology, Hangzhou Dianzi University, Hangzhou 310018 China.
† Department of Computer Science, Illinois Institute of Technology, Chicago, IL60616, USA
‡ SITE, University of Ottawa, Ottawa, Ontario K1N 6N5, Canada

Abstract— **Many TDMA- and CSMA-based protocols try to obtain fair channel access and to increase channel utilization. It is still challenging and crucial in Wireless Sensor Networks (WSNs), especially when the time synchronization cannot be well guaranteed and consumes much extra energy. This paper presents a localized and on-demand scheme ADC to adaptively adjust duty cycle based on *quorum systems*. ADC takes advantages of TDMA and CSMA and guarantees that (1) each node can fairly access channel based on its demand; (2) channel utilization can be increased by reducing competition for channel access among neighboring nodes; (3) every node has at least one rendezvous active time slot with each of its neighboring nodes even under asynchronization. The latency bound of data aggregation is analyzed under ADC to show that ADC can bound the latency under both synchronization and asynchronization. We conduct extensive experiments in TinyOS on a real test-bed with TelosB nodes to evaluate the performance of ADC. Comparing with B-MAC, ADC substantially reduces the contention for channel access and energy consumption, and improves network throughput.**

Index Terms—**Duty Cycle; Quorum Systems; Media Access Control; Wireless Sensor Networks**

I. INTRODUCTION

WSNs have been applied in various environments such as ecological surveillance [1]. Because of hardware limitation, sensor nodes have limited energy and unprecise clocks. Various approaches have been designed to save energy and improve some network performances on throughput, delay and per-node fairness. In order to achieve good cooperation among nodes, synchronization protocols, e.g FTSP [2], were proposed but considerable energy and time were consumed especially in large scale networks as well. How to design protocols to guarantee the communication among nodes under the asynchronous network becomes a very critical and challenging problem. Media Access Control (MAC) protocols let nodes to know when and how to access common channels [3]. Some popular MAC protocols, such as TDMA- and CSMA-based, were designed to share communication medium among nodes by assigning each node some fixed active time slots in TDMA or by letting nodes locally contest their channel access in CSMA. Both of two types of protocols try to build a physically connected network while controlling nodes' active time period in order to reduce energy consumption and improve network throughput.

TDMA has the advantage that time slots are previously scheduled to each node. Therefore, a network can achieve high channel utilization under high media access contention and reduce collision among neighbors with a low cost when their clocks are well synchronized. But TDMA also has some disadvantages [3], some of which are caused by clock asynchronization. B-MAC [4] adopts Low Power Listening (LPL) to solve communication failure caused by clock asynchronization. Although CSMA doesn't austerely require the clock synchronization, it cannot achieve channel utilization as high as TDMA and costs additional time and energy on channel access contention. Thus some hybrid MAC protocols, such as B-MAC [4], S-MAC [3] and T-MAC [5], combining the strengths of both TDMA and CSMA, were proposed. These MAC protocols essentially adopt the LPL technique or improved LPL to allieviate localized asynchronization problem. However, they cannot avoid channel contention and obtain channel allocation fairness in many scenarios [6]. Thus another challenging problem is to decrease the unfair contention for medium access without synchronization while increasing the channel allocation fairness.

This paper designs a localized scheme, named Adjustment of Duty Cycle (**ADC**), based on *Quorum Systems* (QS) [7], to adaptively adjust the duty cycle of each node. A QS is a set of subsets of a universe set such that every pair of subsets intersect with no empty. In recent years, QS is applied to establish control channels in dynamic spectrum access networks [8], to save power [9], and to maximize throughput in limited information multiparty MAC [10]. By **ADC**, each node can select sufficient amount of active time slots, composing a set $\varsigma \subset T$ (called a quorum) according to the amount of its demand while it can sleep to save energy at its rest time in a period. Therefore, its duty cycle $|\varsigma|/|T|$ is adaptively adjusted when the amount of active time slots $|\varsigma|$ is changed. Each node will inform its neighbors of a quorum it selected thus the channel contention among them is decreased. The contributions of this paper are as follows:

• **ADC** can adaptively adjust duty cycle by demand, and

increase the channel allocation fairness comparing with existing contention-based MAC protocols.

• **ADC** guarantees each pair of nodes having sufficient rendezvous active time to implement demand, and the worst case of channel utilization is lower bounded.

• By *the rotation closure property* and *intersection property* of QS, the successful connectivity of a whole network is guaranteed even under asynchronization so no extra energy is consumed on synchronization.

• This paper analyzes the performance of **ADC** under data aggregation, and derives the impact of QS load on network delay, which is defined as a duration from one moment some data is sampled to another that all data is received by the sink.

The organization of this paper is as follows. We first give the network model, formulate our problem and introduce the QS technology in Section II. In Section III, we design our protocol **ADC** and address its preliminary properties. Under clock synchronization and asynchronization, the performances of **ADC** are presented when certain demand is implemented in Section IV and V respectively. In Section VI, we implement our scheme **ADC** in a real test-bed consisting of TelosB nodes and evaluate its performance on real systems. Section VII tells the relative work in recent years. The work of paper is concluded in Section VIII.

II. SYSTEM MODEL AND PRELIMINARY

A. Network Model and Quorum System

We formulate a network by a graph $G(V, E)$, where V (or E) is a set of all nodes (or edges). Let n denote the number of total nodes and s denote an only sink in the network. Each node is assigned a unique ID. A radius of the network G with respect to s, denote by R, is defined as the maximum distance (hops) between s and those nodes in G. This paper studies the duty cycle adjustment under several popular interference models (denoted by **I**): RTS/CTS, protocol model and physical model [11].

QS, denoted by Ω, was used and introduced in precious papers [12][13][8]. A QS $\Omega \subset 2^T$, containing quorums, denoted by Q, is a set of subsets ς of T, where $T = \{\tau_1, \cdots, \tau_m\}$ is a period and composes of m time slots. A *rotation* of a quorum Q is defined as $\mathscr{S}(Q, i) = \{(\tau_j + i) \bmod m | \tau_j \in Q\}$, where i is a non-negative integer. Some QSs satisfy the *rotation closure property*, *i.e.* $\forall i \in \{0, \cdots, m-1\} : Q_1 \cap \mathscr{S}(Q_2, i) \neq \varnothing$, where $Q_1, Q_2 \in \Omega$.

Lemma 1: Grid, torus and cyclic QS all satisfy the *rotation closure property* [13].

B. Problem Statement

Two neighboring nodes u and v can communicate with each other in WSNs *iff* they have at least one rendezvous active time slot. When a network is asynchronous, *i.e.* the clock of each node u has a shift $t_\delta^u \geq 0$ from real time, the set of u's active time slots ς_u accordingly has a shift, *i.e.* $\varsigma_u' = \varsigma_u + \{t_\delta^u\}$. The following equation should be satisfied if a pair of neighboring nodes can communicate with each other under asynchronization.

$$\forall u, v \in V \text{ and } v \in \vartheta_u : \varsigma_u' \cap \varsigma_v' \neq \varnothing, \varsigma_u', \varsigma_v' \subset T \quad (1)$$

where ϑ_u is a *communication set* centered at a node u and a set containing u and those nodes in its communication range. Here, we call a pair of nodes as neighboring when they respectively belong to the communication set of each other. Equation (1) means a pair of neighboring nodes can be physically connected even under asynchronization if their active time slot sets are properly designed.

We define a parameter *demand D* to indicate the amount of data needed to transmit or receive in unit time. Notice that Equation (1) indicates a pair of neighboring nodes should have not only common active time but also enough time to finish all of its demand. This paper aims to deriving a *demand condition* so that each node u can implement its demand by locally choosing a subset $\varsigma_u' \subset T$ to guarantee each pair of nodes in ϑ_u satisfying Equation (1). To obtain the purpose, this paper designs the localized duty cycle adjustment scheme **ADC**.

III. QUORUM SYSTEM BASED TIME SLOT ASSIGNMENT

This section presents our designing of **ADC** and analyzes its properties. These properties indicate **ADC** can achieve better solution than existing protocols on fair medium access even under asynchronization as described in Section I.

A. Designing of ADC

ADC lets each node u obtain a time slot set ς_u so that three following conditions can be satisfied: ① Equation (1); ② Demand condition; ③ The active time is minimized to save energy.

ADC consists of two steps. The first step is to design quorums in a grid QS, denote by Ω_g. In the second step, each node locally selects its quorum base on one-hop information about selected quorums.

At the first step, we construct a grid QS Ω_g with the size $\lceil \sqrt{m} \rceil \times \lceil \sqrt{m} \rceil$ based on a period T as shown in Figure 1. The time slots from T are allocated into the grids of Ω_g from right to left in a row-major manner as shown in Figure 1. In each period, node u requires a set of time slots ς_u to afford of its demand D_u. The relation between the cardinality of ς_u, denoted by κ_u, and D_u is $\kappa_u = \frac{D_u |T|}{\varrho}$, where ϱ is the data rate. Thus we design

Fig. 1. A grid QS Ω_g contains T. There are $\lceil\sqrt{m}\rceil$ rows and columns in Ω_g.

Fig. 2. Q_u and Q_v overlap at 3th line and column.

a quorum Q_u for u and Q_u contains $\lceil\frac{D_u\sqrt{m}}{2\varrho}\rceil$ rows and columns.

Rule 1 [Quorum Design]: The quorums $Q_i \in \Omega_g$ are organized by from i^{th} to $i+\lceil\frac{D_u\sqrt{m}}{2\varrho}\rceil^{th}$ rows and from i^{th} to $i+\lceil\frac{D_u\sqrt{m}}{2\varrho}\rceil^{th}$ columns, where the row and column numbers are labeled as shown in Figure 1 and $i \in \mathbb{Z}+$ and $i \leq \sqrt{m} + 1 - \lceil\frac{D_u\sqrt{m}}{2\varrho}\rceil$.

At the second step, we design a quorum selection method for each node as described in Algorithm 1, in which "a quorum Q_i is occupied" means that $|Q_i \cap Q_j| > \lceil\frac{D_iD_jm}{2\varrho^2}\rceil$, $i \neq j$ when Q_i is selected earlier than Q_j. Figure 2 shows that Q_u is occupied by Q_v if Q_u is selected earlier than Q_v. Notice that the parameter K in Algorithm 1 will be discussed in Lemma 7 and 8. Its value is determined in advance.

Algorithm 1 Quorum Selection

Input: Ω_g, which is allocated time slots from T.
Output: Each node $v \in \vartheta_u$ is allocated a quorum.

1: $v \in \vartheta_u$ sets a positive natural $k = 1$;
2: v sets a list L_o storing the quorums occupied by others.
3: **while** $\vartheta_u \neq \emptyset$ **do**
4:      v randomly selects a quorum Q_i with equal probability, where $i = 1, \cdots, \lceil\sqrt{m} + 1\rceil - \lceil\frac{D_u\sqrt{m}}{2\varrho}\rceil$;
5:      **if** Q_i was occupied and $k \leq K$ **then**
6:          Add Q_i into L_o;
7:          v selects another quorum Q_j, $i \neq j$ and $Q_j \notin L_o$; $k += 1$;
8:      **end if**
9:      Set $k = 1$ and delete v from ϑ_u.
10: **end while**
11: v informs its neighbors of its quorum in a message.

B. Properties of ADC

The properties of **ADC** includes *physical connectivity*, *maximum demand* and *maximum load*. We analyze the effect of **ADC** on network connectivity when the clocks are either synchronous or asynchronous.

The physical connectivity. Physical connectivity is the preliminary condition under which nodes can communicate with each other and implement their demand. In this paper, physical connectivity means the *time-to-rendezvous*, which is the amount of rendezvous slots between arbitrary nodes u and v. If a node $v \in \vartheta_u$ and u are physically connected, the time-to-rendezvous between them should be at least one slot.

By **ADC**, the physical connectivity under both clock synchronization and asynchronization are different. When the clock is synchronous, we can easily obtain Lemma 2. For example, two quorums Q_u and Q_v respectively contain one row and one column in Figure 1. They rendezvous at the time slot 11 and 26. Thus they are physically connected and have two rendezvous active time slots. If one node (for example v) chooses a used quorum Q_u as the instance in Line 5 of Algorithm 1, then two quorums (for example, Q_v and Q_u) have more than two rendezvous time slots.

Lemma 2: If two arbitrary nodes u and $v \in \vartheta_u$ are respectively allocated quorums Q_u and Q_v according to Algorithm 1, they have at least $\lceil\frac{D_uD_vm}{2\varrho^2}\rceil$ rendezvous active time slots when clocks are synchronous.

When clocks are asynchronous, a node u is prone to have clock shift t_δ^u, which is the difference between the local time of u's clock and the exact time. Therefore, the relative clock shift between u and v is $t_\delta(u,v) = t_\delta^u - t_\delta^v$. This paper always assumes $t_\delta^u, t_\delta(u,v) < +\infty$.

Lemma 3: Any pair of quorums in the same QS Ω_g must have at least $\lceil\frac{D_uD_vm}{4\varrho^2}\rceil$ rendezvous active time slots even when the relative clock shift between any pair of nodes is an arbitrary value.

Proof: A grid QS Ω_g satisfies the rotation closure property according to Lemma 1. Thus any two quorums Q_u and Q_v satisfy $Q_u \cap \mathscr{S}(Q_v, i) \neq \varnothing$, where $i = 0, \cdots, m-1$, when $Q_u, Q_v \in \Omega_g$. For any pair of nodes u and v, there is relative clock shift $t_\delta(u,v)$ because of the clock asynchronization. Without loss of generality, let the rotation of Q_v be $\mathscr{S}(Q_v, t_\delta(u,v))$. It means that the slots in Q_v shift because of the relative clock shift. Thus $\mathscr{S}(Q_v, t_\delta(u,v)) \cap Q_u \neq \varnothing$ for $i = 0, \cdots, m-1$ if $v \in \vartheta_u$ since $t_\delta(u,v) \bmod m$ is positive and not bigger than $m - 1$. It means u and v are physically connected because a period totally contains m time slots.

Next we look for a lower bound of the cardinality of $\mathscr{S}(Q_v, t_\delta(u,v)) \cap Q_u$. In **ADC**, any two quorums in the same QS has at least $\lceil\frac{D_uD_vm}{4\varrho^2}\rceil$ rendezvous active time slots according to Theorem 3 of [8] and Lemma 2, *i.e.* $|\mathscr{S}(Q_v, t_\delta(u,v)) \cap Q_u| \geq \lceil\frac{D_uD_vm}{4\varrho^2}\rceil$. ∎

Notice that Lemma 3 is obtained without considering the interference since our scheme **ADC** is applied to MAC. Thus if $|t_\delta(u,v)| \bmod m = 0$ for any pair of nodes u and v, then the physical connectivity of **ADC** under the clock asynchronization is same with that under the clock

synchronization, and the same result can be obtained on the parameter load. In the subsection IV-A, we will analyze the effect of the interference.

Notice that other kind of QSs are also applicable in **ADC** according to Lemma 1 in spite that clocks are synchronous or asynchronous. If a QS should satisfy the rotate closure property, the quorums in the QS should satisfy the condition in Lemma 4.

Lemma 4: If a QS Ω satisfies the rotate closure property, then the cardinality of any quorum in Ω must be more than $\sqrt{m}$ [13].

The maximum demand. Here the demand condition is given so each node u can afford its demand D_u. It is easy to know $D_u \leq \varrho$ if the demand D_u is implementable. When the interference models **I** presented in the subsection II-A are considered, the demand D_u cannot necessarily be close to ϱ. The summation of all nodes in the same communication set ϑ_u should satisfy the condition in Lemma 5 if all of the demand of nodes in ϑ_u can be implemented. Before we give out Lemma 5, we introduce a constant $c_3(\mathbf{I})$ which is determined by the interference model **I**. We calculate $c_3(\mathbf{I})$ by the technique of vertex coloring. The vertex coloring means to color all nodes with minimal number of colors under **I**. Thus nodes with same color are interference-free under **I**.

Lemma 5: When demands of all nodes are implementable, the demands of nodes belonging to the same ϑ_u should satisfy $\sum\limits_{v \in \vartheta_u} D_v \leq \frac{\varrho}{c_3(\mathbf{I})}$, where $c_3(\mathbf{I})$ is a constant related with the interference model **I**.

Proof: When no interference is involved, the nodes in ϑ_u share a period. So $\sum\limits_{v \in \vartheta_u} \kappa_v \leq T \Rightarrow \sum\limits_{v \in \vartheta_u} \frac{D_v T}{\varrho} \leq T \Rightarrow \sum\limits_{v \in \vartheta_u} D_v \leq \varrho$. Under the interference model **I**, each communicate set can transmit or receive in every $c_3(\mathbf{I})$ periods in order to be interference-free. The average data rate is $\frac{\varrho}{c_3(\mathbf{I})}$. So $\sum\limits_{v \in \vartheta_u} D_v \leq \frac{\varrho}{c_3(\mathbf{I})}$. ∎

We find it is not always suitable to decrease the maximum demand of each node since a node should keep active in at least $\sqrt{m}$ time slot to satisfy the rotate closure property according to Lemma 4. Hence demand of a node u should have a lower bound. Since the Q_u contains $\lceil \frac{D_u \sqrt{m}}{2\varrho} \rceil$ rows and a column, the lower bound can be obtained by Lemma 6.

Lemma 6: When the Ω_g satisfies the rotation closure property, the demand of each node should satisfy $D_u \geq c_1 \varrho$, where $c_1 = 1 - \sqrt{1 - 1/\sqrt{m}}$, when $m > 1$.

Proof: For an arbitrary node u, the cardinality of Q_u is $2\sqrt{m}\lceil \frac{D_u \sqrt{m}}{2\varrho} \rceil - \lceil \frac{D_u \sqrt{m}}{2\varrho} \rceil \times \lceil \frac{D_u \sqrt{m}}{2\varrho} \rceil$. By Lemma 4, we have $2\sqrt{m}\lceil \frac{D_u \sqrt{m}}{2\varrho} \rceil - \lceil \frac{D_u \sqrt{m}}{2\varrho} \rceil \times \lceil \frac{D_u \sqrt{m}}{2\varrho} \rceil \geq \sqrt{m}$. $\Rightarrow 2\lceil \frac{D_u \sqrt{m}}{2\varrho} \rceil - \lceil \frac{D_u}{2\varrho} \rceil \times \lceil \frac{D_u \sqrt{m}}{2\varrho} \rceil \geq 1$. When $m = 1$, the above inequality can be always satisfied because $D_u/\varrho \leq 1$. When $m > 1$, $D_u \geq c_1 \varrho$, where $c_1 = 1 - \sqrt{1 - 1/\sqrt{m}}$. ∎

Let $D_k = \min\limits_{v \in \vartheta_u} D_v$ so $D_k \geq c_1 \varrho$ according to Lemma 6. Thus we have $\sum\limits_{v \in \vartheta_u} D_v \geq c_1 |\vartheta_u| \varrho$.

The load bound. When the condition in Lemma 5 is satisfied, there still exists competition between a pair of nodes u and v in the same ϑ, where the competition between them because $|Q_u \cap Q_v| > \lceil \frac{D_u D_v m}{2\varrho^2} \rceil$, $u \neq v$, as described in Algorithm 1. Notice that the parameter K in Algorithm 1 is used to decrease the competition, where K is the number of times in which the same node selects different quorums. It is important to take full advantage of the time diversity of medium access, *i.e.* to decrease the competition between different quorums, in order to control the channel congestion. Some previous work designed protocol to minimize the load [8]. But it is not suitable under a more practical case in this paper. That is each node has demand different from others' because of their different network tasks. Furthermore, we show that it cannot fully use the time slots when the demand is very low, under which it will degrade the channel utilization to decrease the load. In **ADC**, we present the upper- and lower-bound of the load under the demand constraint given in Lemma 5 and 6.

We first give out a lower bound of the quorum load. Two propositions 4.1 and 4.2 in [14] gave a result that $\mathcal{L}(\Omega) \geq \max\{\frac{1}{c_4(\Omega)}, \frac{c_4(\Omega)}{m}\}$, where $c_4(\Omega)$ is the size of smallest quorum in Ω, and $c_4(\Omega) \geq \sqrt{m}$ according to Lemma 6. We then have $\frac{1}{c_4(\Omega)} \leq \frac{1}{\sqrt{m}}$ and $\frac{c_4(\Omega)}{m} \geq \frac{1}{\sqrt{m}}$. Therefore, $\mathcal{L}(\Omega) \geq \frac{c_4(\Omega)}{m}$, which is different from the result in Theorem in [8], because the rotation closure property is considered.

Next we discuss $c_4(\Omega)$. Notice that the cardinality of Q_u should be not less than $\sqrt{m}$ when the demand of each node $D_u \geq c_1 \varrho$. Otherwise, the rotation closure property cannot be satisfied. Thereinafter, we analyze the bound of the load in two case: $K = 1$ and $K \geq 2$.

In Lemma 5, an obvious upper bound of the QS load appears when a node is required to afford the full demand. *i.e.*, the demand D_v of a node $v \in \vartheta_u$ is not less than the maximal data rate, $D_v \geq \varrho$. Under this case, the cardinality of each quorum is m. Because each node randomly selects a quorum with equal probability and the probability that each time slot is included in a quorum is $\frac{1}{|\vartheta|}$, $\mathcal{L}_S(i) = \sum\limits_{Q \in \Omega: \tau_i \in Q} P_S(Q) = \sum\limits_{Q \in \Omega: \tau_i \in Q} \frac{1}{|\vartheta|} = 1$, $\mathcal{L}_S(\Omega) = 1$. Under **ADC**, the cardinality of each quorum is not bigger than m and the probability that each time slot is included in a quorum is less than $\frac{1}{|\vartheta|}$. So $\mathcal{L}_S(\Omega) \leq 1$.

Lemma 7: In **ADC**, the load of Ω_g is less than $\frac{c_5}{|\vartheta_u|}$, *i.e.*, $\mathcal{L}_S(\Omega_g) \leq \frac{c_5}{|\vartheta_u|}$, when the QS Ω_g satisfies rotation closure properties and $K = 1$, where $c_5 = \lceil \frac{1}{c_3} \rceil - \lceil \frac{1}{4c_3^2} \rceil$.

Proof: Since the cardinality of a quorum Q_v ($v \in \vartheta_u$) is $|Q_v|$ under **ADC**, the probability that a time slot is included in Q_v is $\frac{|Q_v|}{m}$. Suppose there are γ quorums in each Ω. $|\vartheta_u| \le \gamma \le \sqrt{m}$ when the summation of all nodes' demand in the same ϑ_u is implementable. So the probability that a quorum is chosen by a node is $\frac{1}{\sqrt{m}} \le \frac{1}{\gamma} \le \frac{1}{|\vartheta_u|}$. Therefore, the probability that each time slot is included in the quorum Q_v when there are $|\vartheta_u|$ quorums is $\frac{1}{|\vartheta_u|} \times \frac{|Q_u|}{m}$. The load induced by the strategy $\mathcal{S}$ on a time slot τ_i is $\mathcal{L}_{\mathcal{S}}(\tau_i) = \sum\limits_{Q_u \in \Omega_g : \tau_i \in Q_u} \frac{1}{|\vartheta_u|} \times \frac{|Q_u|}{m}$. Notice there are $|\vartheta_u|$ nodes to select quorums from Ω_g, *i.e.* $\Omega_g = \{Q_v : Q_v \in \vartheta_u\}$. So

$$
\begin{aligned}
\mathcal{L}_{\mathcal{S}}(\tau_i) &= \frac{1}{\gamma m} \sum_{Q_u \in \Omega_g : \tau_i \in Q_u} |Q_u| \\
&\le \frac{1}{|\vartheta_u| m} \sum_{\tau_i \in Q_u \in \Omega_g} \{2\sqrt{m}\lceil\frac{D_u\sqrt{m}}{2\varrho}\rceil - \lceil\frac{D_u\sqrt{m}}{2\varrho}\rceil^2\} \\
&= \frac{1}{|\vartheta_u|} \sum_{\tau_i \in Q_u \in \Omega_g} \{\lceil\frac{D_u}{\varrho}\rceil - \lceil\frac{D_u}{2\varrho}\rceil^2\} \le \frac{1}{|\vartheta_u|}(\lceil\frac{1}{c_3}\rceil - \lceil\frac{1}{4c_3^2}\rceil)
\end{aligned}
$$
(2)

Thus the load induced by the strategy $\mathcal{S}$ on the quorum system Ω_g is $\mathcal{L}_{\mathcal{S}}(\Omega_g) = \max\limits_{\tau_i \in T} \mathcal{L}_{\mathcal{S}}(\tau_i) \le \frac{1}{|\vartheta_u|}(\lceil\frac{1}{c_3}\rceil - \lceil\frac{1}{4c_3^2}\rceil) = \frac{c_5}{|\vartheta_u|}$, where $c_5 = \lceil\frac{1}{c_3}\rceil - \lceil\frac{1}{4c_3^2}\rceil$. ∎

Now we analyze the upper bound of the QS load when $K \ge 2$. Here we treat a quorum containing a row and a column and call the quorum as a bin, so each node actually selects several such bins according to the *Rule 1*. Each node has K times to select its quorum in Algorithm 1. Thus the quorum selection problem in Algorithm 1 is equivalent to the *k-round ball placement problem* [15]. When $m_i = 1$, we can obtain that the maximum load achieved by Algorithm 1 is less than $\frac{\log\log\sqrt{m}}{\log K}$ *w.h.p* according to Theorem 6 of [16]. When $m_i \ge 2$ in each round, *i.e.* several bins are selected together in each round, it is equivalent to combining several bins into one. Therefore, the total number of bins is correspondingly reduced. We can obtain Lemma 8.

Lemma 8: The maximum load achieved by Algorithm 1 is less than $\frac{\log\log\sqrt{m}}{\log K}$ *w.h.p.*

IV. SYNCHRONOUS DEMAND IMPLEMENTATION

This section evaluates the performance of **ADC** when the data aggregation is implemented and clocks are synchronous, which we call as synchronous demand implementation. In order to implement the demands, we construct a tree and design the specific demand implementation methods. The performance of **ADC** under the asynchronization will be analyzed in the next section by applying the results of this section.

A. Tree Construction

Firstly, a tree $\mathscr{T}$ is constructed based on G by constructing a connected dominator set (CDS). we then define a new conception *region* in order to obtain a conflict-free quorum assignment for each node.

We construct a CDS by the breadth-first-search (BFS) based on G. Each dominatee connects with the dominator closest to it. In this way, $\mathscr{T}$, rooting at s, can be constructed and is ranked into $\mathscr{L}$ levels from s. The level of s is labeled l_0. The parent and the children of a node u are denoted by $p(u)$ and $c(u)$ respectively.

We assign QSs for each region in two phases. At the first phase, a new conception *region* is defined to obtain the conflict-free partition by the vertex coloring. At the second phase, we assign each region with a period so nodes can be active without confliction.

Phase I: After $\mathscr{T}$ is constructed, each node can know its own level and its one-hop neighbors' IDs and levels. The one-hop neighborhood of each node u is denoted by $N_1(u)$ and notice that $u \in N_1(u)$. We call a one-hop neighborhood of a non-dominatee node as a *region* (denoted by σ) in the tree $\mathscr{T}$. Notice that any dominatee does not form a region. Because of interference within a network, the QSs of some neighboring regions cannot be assigned a same time slots set. Here, we say two regions σ_1 and σ_2 are *neighboring* (or *over-lap*) if there are two nodes $u \in \sigma_1$ and $v \in \sigma_2$ and u (or v) locates in the interference range of v (or u).
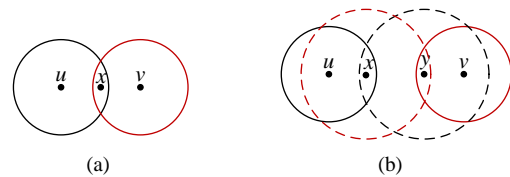

Fig. 3. (a) σ_u and σ_v are two over-lap regions. (b) σ_u and σ_v are two neighboring regions.

If two regions are conflict-free, *i.e.* they are not neighboring and over-lap, we color them with a same color. So it is vertex coloring problem to find the minimal number of colors. The least number of colors, denoted by c, necessary to color all regions, is affected by the interference model **I**, *i.e.* we need at least $c(\mathbf{I})$ to color all regions and the regions with same color are conflict-free. We label each region σ_u with a color index θ_{σ_u}, $\theta_{\sigma_u} \in \mathscr{C} = \{1, \cdots, c(\mathbf{I})\}$. In order to determine the least number of colors, we define a parameter (denoted by φ) to denote the interference range under different interference models. Thus any pair of regions can have same color if they are more than η hops apart.

Phase II: Allocate each region with a QS. Because the quorum is designed in the subsection III-A, each node belongs to at least one region and some belong to several

regions. For example, x belongs to two regions σ_u and σ_v in Figure 3(a). Thus the number of quorums each node occupies is same with the number of regions it belongs to. We assign each region with a period,*i.e.* a QS, according to the color so each neighboring or over-lap regions can be conflict-free. For example, σ_u and σ_v are respectively assigned two periods T_u and T_v. Let $T_u = \{1, 2, 3\}$ and $T_v = \{4, 5, 6\}$ so $T_u \cap T_v = \varnothing$.

We use a natural number $i(i \in \mathbb{Z})$ to label the ID of a region in order to assign the time slot set conveniently. Firstly, we color all the regions by Algorithm 2. Secondly, each region i is assigned a slot set $\mathfrak{L}_i$ by Algorithm 3.

Algorithm 2 Region Coloring

Input: The labels of all regions, σ_i, $i = 1, \cdots, |CDS|$ and the color set $\mathscr{C}$.

Output: The colored regions.

1: **for** $i = 1, \cdots, |CDS|$ **do**
2:     **if** There is no region colored within η hops centered at a region σ_i **then**
3:       σ_i labels itself with a color θ_{σ_i} ($\in \mathscr{C}$).
4:     **else**
5:       σ_i selects a color θ_{σ_i} from $\mathscr{C}$ and θ_{σ_i} is different from the colors of other nodes within η hops centered at σ_i's.
6:     **end if**
7: **end for**

Algorithm 3 Slot Set Assignment

Input: All the colored regions, θ_{σ_i}, $i = 1, \cdots, |CDS|$ and $\theta_{\sigma_i} \in CDS$.

Output: Each region σ_i obtains a slot set $\mathfrak{L}_{\sigma_i}$.

1: **for** $j = \mathscr{L}, \cdots, 0$ and $\sigma_i \subset l_j$ **do**
2:     **for** $i = 1, \cdots, |CDS|$ **do**
3:       A region σ_i^j colored with a color index $\theta_{\sigma_i^j}$ is assigned the slot set $\mathfrak{L}_{\sigma_i^j} = \theta_{\sigma_i^j} \ mod \ \varphi \times \kappa$.
4:       **if** $\mathfrak{L}_{\sigma_i^j} > \max\limits_{all \ c(i)} \mathfrak{L}_{\sigma_{c(i)}^{(j+1)}}$ **then**
5:         $\mathscr{S}(\mathfrak{L}_{\sigma_i^j}, m)$.
6:       **end if**
7:     **end for**
8: **end for**

After each region is assigned a period, it can obtain its quorum according to Algorithm 1. We can design a determinate quorum selection method rather than the random one in Algorithm 1. The determinate quorum selection method is given in Algorithm 4. The difference between Algorithm 1 and Algorithm 4 is that each node doesn't select quorums randomly. By Algorithm 4, we can obtain some properties.

Here we give a notion: *logical connection*. Nodes u and v are logically connected if they locate in each other's transmission range. If u and v cannot be active in common time slots, they cannot communicate with each other. A graph can be logically connected by topology control algorithms. Thus $\mathscr{T}$ is logically connected if G is. According to Algorithm 4, it is easy to know that each pair of nodes are physically connected in $\mathscr{T}$ if G is logically connected.

Algorithm 4 Determinate Quorum Selection

1: Each node u collects the level label and quorums of its neighbors in $N_1(u)$;
2: u classifies its neighbors into three sets: S_1, S_2 and S_3. S_1 contains the node in l_{k-1} and S_2 contains the nodes in l_k and S_3 contains the nodes in l_{k+1};
3: **if** u is the sink **then**
4:     $S_1 = \emptyset$;
5: **end if**
6: **for** $i = 1, \cdots, w$ **do**
7:     **if** Q_i is not chose **then**
8:       According to the order S_3, S_2 and S_1, each node in S_3, S_2 and S_1 chooses Q_i.
9:     **end if**
10: **end for**

Lemma 9: Each pair of neighboring or over-lap regions, and each pair of links in a same region are conflict-free according to Algorithm 4.

Proof: According to Algorithm 2, any neighboring (or overlap) regions are colored different colors. In Algorithm 3, the regions σ_i and σ_j colored different colors are assigned different slot set $\mathfrak{L}_{\sigma_i}$ and $\mathfrak{L}_{\sigma_j}$. $\mathfrak{L}_{\sigma_i} \cap \mathfrak{L}_{\sigma_j} = \varnothing$ according the line 3 in Algorithm 3. In Algorithm 4, each node is assigned a quorum different from that of others in the same region. According to the definition of QS, there are only two nodes to be active simultaneously. ∎

Lemma 10: If a node u (expect s) and its parent $p(u)$ are respectively assigned the time slots τ_u and $\tau_{p(u)}$, then $\tau_u < \tau_{p(u)}$ according to Algorithm 4.

In Algorithm 3, the active period of each node is earlier than that of its parent. According to Algorithm 4, the node in the level l_i is active earlier than that in l_{i-1} in the same region. Thus we obtain Lemma 10. It is easy to obtain that each parent will transmit after it receives all packets from all of its children in the same period.

B. Data Aggregation

This section discusses performance of **ADC** under data aggregation. Under data aggregation, the demand of a parent is the summation of its children, *i.e.* $D_{p(u)} = \sum\limits_{u \in \sigma_{p(u)}} D_u$. We can obtain that the maximal size of a QS

among all QSs is determined by the maximal degree of $\mathscr{T}$ so we have Lemma 11.

Lemma 11: $\max |\Omega| = \max\limits_{u \in \mathscr{T}} d_u^2$.

The delay of data aggregation is given in Theorem 12.

Theorem 12: The maximal delay of data aggregation is $O(\varphi m R + \Delta^2)$ by Algorithm 2 and 4.

Proof: Each region σ_i is assigned a QS Ω_i so it costs $|\Omega_i|$ to finish all the transmission in σ_i. Since $|\Omega_i| \leq \max |\Omega|$ according to Lemma 11, $\max |\Omega| = \max\limits_{u \in \mathscr{T}} d_u^2 = \Delta^2$. There exists an assignment method such that the number of different color is at most φ in each level of the tree $\mathscr{T}$. Let each region σ_i be assigned a slot set $\mathfrak{L}_{\sigma_i}$. Thus it needs at most $\varphi \mathfrak{L}_{\sigma_i}$ in each level. Because there are totally $\mathscr{L}$ levels and the lower level l_i is active earlier than the higher level l_j ($i > j$), the sink costs $\varphi \mathfrak{L}_{\sigma_i} \mathscr{L}$ time to collect all data. Because $\mathfrak{L}_{\sigma_i} \leq m$, $\mathscr{L} \leq R$ and $\mathfrak{L}_{\sigma_i} \leq m$, $\varphi \mathfrak{L}_{\sigma_i} \mathscr{L} \leq \varphi m R$, the maximal delay of the data aggregation by our method is $\varphi m R + \Delta^2$. ∎

Any schedule has delay at least R (or D), where D is the radius of a network [17]. When s locates at the center of a topology, the delay lower boundary can be reduced to be $(\varphi m + 1)D + \Delta^2$, where $D = R/2$ [17].

V. ASYNCHRONOUS DEMAND IMPLEMENTATION

ADC does not require the global clock synchronization. This section aims to analyze the delay of data aggregation under asynchronization. Existing algorithms are designed to bound the delay, such as time slot assignment algorithm in [18]. However, neighboring nodes may not be physically connected under asynchronization. Thus an additional method is given to ensure each pair of neighboring nodes are physically connected in the subsection V-B.

A. Asynchronous Delay of Data Aggregation

We assume that the clock shift $t_\delta^u(t)$ of a node u randomly and uniformly distributes in the interval $[-\infty, \infty]$, where t^u and t are local time and exact time. When u selects a quorum Q_u, u is actually active in $\mathscr{S}(Q_u, t_\delta(t))$ because of the clock shift t_δ. According to Algorithm 3, every region is assigned a period in every φ periods. Without loss of generality, $u \in \sigma_u$ is active in a period $T_{u+i\varphi}$, where $i = 1, 2, \cdots$. For a pair of neighboring nodes u and v, the relative clock shift between them is $t_\delta^{uv}(t) = t^u - t^v$. Thus u and v have common active time slots only if $\mathscr{S}(Q_v, t_\delta^{uv}(t)) \cap Q_u \neq \varnothing$, where $Q_u \in T_{u+i\varphi}$. We have the following lemma.

Lemma 13: If u and v locate in the same region and $i - 1 < \frac{t_\delta^{uv}(t)}{T\varphi} < i + 1$, where $i = 1, 2, \cdots$, u and v are physically connected.

According to Lemma 13, nodes may not be physically connected because the data aggregation scheme is adopted under the clock asynchronization. Notice that it is not caused by **ADC**.

Asynchronous clock causes additional delay on the delay of data aggregation in order to ensure each pair of u and v could have common active time slots to communicate with each other in T_u under asynchronization. That means u and v have to postpone their communication because of the clock shift t_δ^{uv}. But they can communicate with each other within at most $\varphi - 1$ additional periods delay if they are physically connected according to Lemma 13. Therefore, we can obtain the delay of data aggregation based on Theorem 12 as illustrated in the following lemma.

Lemma 14: The delay of data aggregation is $O((2\varphi - 1)TR + \Delta^2)$ under asynchronization if each pair of neighboring nodes are physically connected.

B. Quorum Share

Although Lemma 3 ensures each pair of nodes (including non-neighboring) are physically connected, Lemma 13 indicates each pair of neighboring nodes must be unable to communicate with each other in some periods since Algorithm 3 assigned each region with discontinuous periods. This section designs a scheme to solve this problem. Suppose σ_u is assigned periods T_i, $i = 0, \varphi, 2\varphi, \cdots$, and the clock shift of a node $v \in \sigma_u$ is t_δ^v and Q_v contains the time slots set ς_v. When $\varsigma_v' = \varsigma_v + \{t_\delta^v\}$ locates in the periods which does not satisfy the inequality in Lemma 13, v conflicts with some nodes, *i.e.* the quorum Q_v shifts into the QS of some other nodes x_i, $i = 1, 2, \cdots, \leq |\vartheta_x|$, in another region σ_x, where v locates in the interference range of x_i. Notice that there are two kinds of quorum shifting: (1) a quorum Q_v only shifts between periods when $t_\delta \bmod |T| = 0$; (2) a quorum Q_v shifts among a QS when $t_\delta \bmod |T| \neq 0$.

In our scheme described in Algorithm 5, v and x_i, $i = 1, 2, \cdots, \leq |\vartheta_x|$ share the quorum Q_v with equal probability. Our scheme can deal with two kinds of quorum shifting.

Algorithm 5 Quorum Share

1: Each node v sets a list L_t storing the nodes' ID, which occupy v's quorum Q_v.
2: **while** v detects that Q_v is occupied by the nodes not in its one-hop neighborhood. **do**
3:    v sets a positive natural number k to be $k+ = 1$;
4:    v sets itself to be active in periods T_i, where $i = 0, k\varphi, 2k\varphi, \cdots$;
5: **end while**

By Algorithm 5, each pair of neighboring nodes are physically connected even when the time slot allocation

algorithm is implemented in Algorithm 3. According to Algorithm 5, u would be active in T_i, where $i = 0, k\varphi, 2k\varphi, \cdots$. It means that u is active in part of periods. Since the grid QS satisfies the rotation closure property, u is still physically connected with its neighboring nodes. Therefore, we can obtain Lemma 15.

Lemma 15: A pair of neighboring nodes can be physically connected by Algorithm 5 under asynchronization.

Proof: Suppose a pair of neighboring nodes u and v respectively select quorums Q_u and Q_v. Thus $Q_u \cap Q_v \neq \varnothing$. Q_u and Q_v respectively contain the time slot set ς_u and ς_v. Denote the relative clock shift between u and v is t_δ^{uv}. Thus $\varsigma_u' = \varsigma_u + \{t_\delta^{uv}\}$. Since the grid QS, $\varsigma_u' \cap \varsigma_v \neq \varnothing$. ∎

In Algorithm 5, each node u would be active in part of periods under asynchronization. We find that the whole network delay is prolonged while each pair of neighboring nodes are guaranteed to be physically connected. The demand of u would be implemented lingeringly because the quorum Q_u of u moves to Q_u' in another period as shown in Figure 4. We suppose Q_u' locates in the period T_1 and T_1 is originally assigned to the nodes in the region σ_1. So u would share the same quorum with some nodes in σ_1. At the worst case, all nodes in σ_1 shunt one turn, *i.e.* they are active in $0, 2\varphi, 4\varphi, \cdots$. The delay caused by the clock shift is at most t_δ^u periods when there is only u which has clock shift under the interference model **I**. When the clocks of every nodes shift, their quorum may also shift. For example, Q_i and Q_j respectively move to new places, such as Q_i' and Q_j', which locate in different periods in Figure 4. The worst case is that $\varphi - 1$ regions shift into one regions, thus the additional delay is at most $\varphi(\varphi - 1)m$. According to Theorem 12, we have the following lemma.

Lemma 16: By Algorithm 5, the delay on the data aggregation is at most $O(\varphi^2 mR + \Delta^2)$ when the clocks are asynchronous.
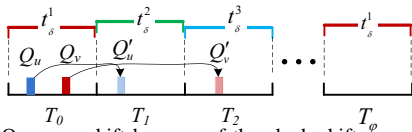

Fig. 4. Quorums shift because of the clock shift.

VI. PERFORMANCE EVALUATION

This paper evaluated **ADC** and B-MAC in a real testbed running TinyOS on TelosB motes. The testbed composes of one hundred nodes. We compare the performance **ADC** against B-MAC on the throughput and packet receiving ratio (PRR).

A. Experiment Setup

We randomly deployed 100 nodes on an in-door testbed. Each sensor node works with its modified internal antenna, the transmission range of which can be as small as 10cm. Thus nodes in the original network can still communicate with each other by multi-hop. After deployment, we start our experiment, composing of two phases. At first phase, all the nodes are initially set with 100% duty cycle. At the second phase, nodes in a same region selected their quorums according to their locations (parent or leaf node) in the tree and the number of the leaf nodes under **ADC**. The duty cycle is set 20% under B-MAC.

Under **ADC**, each Ω contains 100 time slots, *i.e.* $m = 100$. Each time slot is respectively set as $50ms$, $1s$, $2s$ and $5s$. Each node samples data in every $20ms$, $50ms$, $100ms$, $200ms$, $300ms$, $500ms$, $800ms$, $1s$, $1.5s$ and $2s$, which are called as the data generation period in Figure 5 and 6. When the experiment starts, the sink broadcasts a message to synchronize the clocks of all nodes.

B. Performance Comparison

In this section, we compare the performance of B-MAC and **ADC** on the network throughput and PRR. Although we care about the channel utility, fairness and energy consumption, the experiment results on two parameters, throughput and PRR, synthetically reflect the channel utility.

Throughput. Figure 5 shows the network throughput respectively under **ADC** and B-MAC. Each node except the sink generates data at different rate. Because the nodes should compete the channel access when transmitting each packet, much time is wasted. When the time slot size is big, for example, 1s, 2s and 5s, the nodes should cost time on the channel access competition and any pair of neighboring nodes have much continual time to transmit packets. As shown in Figure 5(b), 5(c) and 5(d), the throughput under B-MAC is much lower than that under **ADC** when the time slot size is bigger, such as 1s, 2s and 5s. Although the network is synchronized at the right beginning of the experiment, the clocks of all nodes shift off after a period of time. Some of nodes scheduled to wake up at common time may mismatch especially when time slots are set to be very short. The throughput under B-MAC is litter higher than that under **ADC** when the time slots size is small, such as $50ms$. Notice that the network throughput under **ADC** does not change much when the time slot size changes. But the time slot size has much effect on the throughput under B-MAC. The throughput under both B-MAC and **ADC** decrease with the increasing of the data generation period when the period is higher than $900ms$.

PRR. PRR reflects the channel utility within a network. The PRRs under both B-MAC and **ADC** increase with the increasing of the data generation period. When the time slot size is big, such as, 1s, 2s and 5s, the
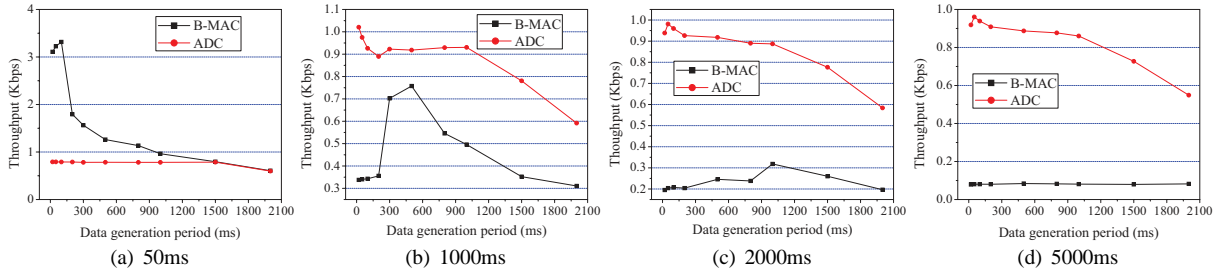
Fig. 5. The network throughput respectively under **ADC** and B-MAC with different data generation periods.
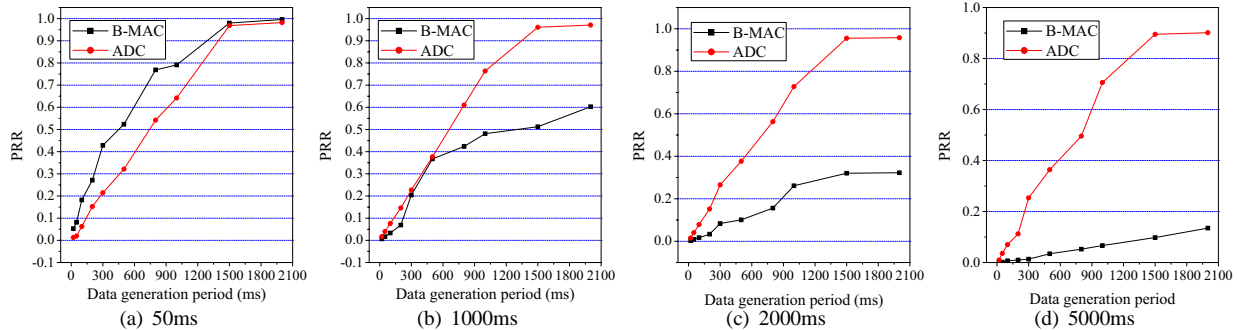

Fig. 6. The network PRR respectively under **ADC** and B-MAC with different data generation periods.

PRR under **ADC** is much higher than that under B-MAC. The results are similar to those on the network throughput. The time slots size has much effect on the throughput under B-MAC instead of that under **ADC**. The PRR under B-MAC increases with the decreasing of the time slot size.

VII. RELATED WORK

A. Duty Cycle

In WSNs, many works were put on duty-cycled networks as following. [19] designed a data forwarding technique to optimize the data delivery ration, end-to-end delay or energy consumption under low-duty-cycle by synchronized mode. [20] designed an opportunistic flooding scheme for low-duty-cycle networks with unreliable wireless links and predetermined wording schedules by locally synchronization. [21] provided a benchmark for assessing diverse duty-cycle-aware broadcast strategies and extend it to distributed implementation. [22] minimized broadcast transmission delay by a set-cover-based approximation scheme with both centralized and distributed algorithms. Using the β-synchronizer, a fast distributed algorithm built all-to-one shortest paths with polynomial message and time complexity [23]. [24] designed an asynchronous duty-cycle broadcasting to let a node be active very long time when it need broadcast the data to a large number of neighbors. [25] analyzed the performance of geographic routing over duty-cycled nodes and presented a sleeping scheduling

algorithm that can be tuned to achieve a target routing latency. [26] presented an alternative frame-let based LPL implementation to improve the network performance by opportunistically aggregating packets over the radio channel.

B. MAC protocol

Some protocols were designed to combine the advantages of TDMA and CSMA. [27] proposed a hybrid MAC protocol, called Z-MAC, in which, a node always performs carrier-sensing before transmission. Z-MAC consumes much energy on the carrier-sensing and also needs local synchronization among senders in two-hop neighborhoods. S-MAC [3] and T-MAC [5] employ RTS/CTS mechanism to solve the the synchronization failure. Since these protocols use RTS/CTS, their overhead is quite high [27]. B-MAC [4] is the default MAC in the operate system of Mica2 and adopts Low Power Listening (LPL) to solve the asynchronization. Since LPL consumes much energy, X-MAC reduces the energy consumption and latency by employing short preamble and embedding address information of the target in the preamble [28]. So the non-target receivers can quickly go back to sleep and the energy is saved. LPL based preamble transmission may occupy the medium for much longer time than actual data transmission. So [29] designed an asynchronous duty cycle MAC: RI-MAC. It wastes energy especially under low traffic load and the interference is increased because of the periodical

broadcasting of beacons.

MAC protocols are also designed to reduce energy consumption, such as S-MAC [3] and T-MAC [5]. [30] considered LPL approaches, such as WiseMAC and B-MAC, are limited to duty cycles of 1-2% and designed a new MAC protocol called scheduled channel polling (SCP) to ensure that duty cycles of 0.1% and below are possible. [30] dynamically adjusts duty cycles in the face of busy networks and streaming traffic in order to reduce the latency. [31] presented a new receiver-initiated link layer A-MAC to support multiple services under a unified architecture more efficiently and scalably than prior designs. [32] designed a TDMA-based MAC primitive module PIP to achieve high throughput for reliable bulk data transfer.

VIII. CONCLUSION

Energy conservation is a fundamental issue in WSNs, which usually relies on wise designs of duty cycling mechanisms. In this paper, we propose a localized scheme, **ADC**, to adaptively adjust the duty cycles of all nodes in WSNs. **ADC** leverages the technique of QS and adjust the duty cycles of sensor nodes according to their demand, so that all nodes can fairly access their common channels. We address both synchronous and asynchronous cases with **ADC** and implement it on a test-bed with 100 TelosB nodes. The results demonstrate that ADC significantly improves the WSN performance such as network throughput and PRR. In our future work, we plan to design protocols of duty cycle adjustment, which has more high utilization of active time and lower duty cycle, so the energy consumption efficiency can be increased.

REFERENCES

[1] L. Mo, Y. He, Y. Liu, J. Zhao, S.J. Tang, X.Y. Li, and G. Dai. Canopy closure estimates with greenorbs: Sustainable sensing in the forest. In *ACM Sensys*, pages 99–112, 2009.

[2] M. Maróti, B. Kusy, G. Simon, and Á. Lédeczi. The flooding time synchronization protocol. In *ACM Sensys*, pages 39–49, 2004.

[3] W. Ye and J. Heidemann. Medium access control in wireless sensor networks. *Wireless sensor networks*, pages 73–91, 2004.

[4] J. Polastre, J. Hill, and D. Culler. Versatile low power media access for wireless sensor networks. In *ACM Sensys*, pages 03–05, November 2004.

[5] T. Van Dam and K. Langendoen. An adaptive energy-efficient MAC protocol for wireless sensor networks. In *ACM Sensys*, page 180, 2003.

[6] Y. Jian and S. Chen. Can CSMA/CA networks be made fair? In *Proceedings of ACM MobiCom*, pages 235–246, 2008.

[7] D. Malkhi and M. Reiter. Byzantine quorum systems. *Distributed Computing*, 11(4):203–213, 1998.

[8] K. Bian, J.M. Park, and R. Chen. A quorum-based framework for establishing control channels in dynamic spectrum access networks. In *Proceedings of ACM MobiCom*, pages 25–36, 2009.

[9] S.H. Wu, M.S. Chen, and C.M. Chen. Fully adaptive power saving protocols for ad hoc networks using the Hyper Quorum System. In *IEEE ICDCS*, pages 785–792, 2008.

[10] P. Chaporkar, S. Sarkar, and R. Shetty. Dynamic quorum policy for maximizing throughput in limited information multiparty MAC. *IEEE/ACM Transactions on Networking (TON)*, 14(4):848, 2006.

[11] S. Li, Y. Liu, and X.Y. Li. Capacity of large scale wireless networks under Gaussian channel model. In *Proceedings of ACM MobiHoc*, pages 140–151, 2008.

[12] 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*, 43(3):317–337, 2003.

[13] J.R. Jiang, Y.C. Tseng, C.S. Hsu, and T.H. Lai. Quorum-based asynchronous power-saving protocols for IEEE 802.11 ad hoc networks. *Mobile Networks and Applications*, 10(1):169–181, 2005.

[14] M. Naor and A. Wool. The load, capacity, and availability of quorum systems. In *ANNUAL SYMPOSIUM ON FOUNDATIONS OF COMPUTER SCIENCE*, volume 35, pages 214–214, 1994.

[15] X.Y. Li, Y. Wang, and W. Feng. Multiple Round Random Ball Placement: Power of Second Chance. In *INFOCOM*, page 448, 2009.

[16] M. Mitzenmacher. The power of two choices in randomized load balancing. *IEEE TPDS*, pages 1094–1104, 2001.

[17] ShiGuang Wang ShaoJie Tang GuoJun Dai JiZhong Zhao Yong Qi Xiang-Yang Li, XiaoHua Xu. Efficient Data Aggregation in Multi-hop Wireless Sensor Networks under Physical Interference Model. In *Proceedings of IEEE MASS*, 2009.

[18] Peng-Jun Wan, Scott C.-H. Huang, Lixin Wang, Zhiyuan Wan, and *etc*. Minimum-latency aggregation scheduling in multihop wireless networks. In *Proceedings of ACM MobiHoc*, pages 185–194, 2009.

[19] Y. Gu and T. He. Data forwarding in extremely low duty-cycle sensor networks with unreliable communication links. In *ACM Sensys*, page 334. ACM, 2007.

[20] S. Guo, Y. Gu, B. Jiang, and T. He. Opportunistic flooding in low-duty-cycle wireless sensor networks with unreliable links. In *ACM MobiCom*, pages 133–144, 2009.

[21] F. Wang and J. Liu. Duty-cycle-aware broadcast in wireless sensor networks. In *IEEE INFOCOM*, 2009.

[22] J. Hong, J. Cao, W. Li, S. Lu, and D. Chen. Minimum-Transmission Broadcast in Uncoordinated Duty-Cycled Wireless Ad Hoc Networks. *IEEE TVT*, 59(1):307–318, 2010.

[23] S. Lai and B. Ravindran. On Distributed Time-Dependent Shortest Paths over Duty-Cycled Wireless Sensor Networks. In *IEEE INFOCOM*, 2010.

[24] Y. Sun, O. Gurewitz, S. Du, L. Tang, and D.B. Johnson. ADB: an efficient multihop broadcast protocol based on asynchronous duty-cycling in wireless sensor networks. In *ACM Sensys*, pages 43–56, 2009.

[25] S. et al Nath. Communicating via fireflies: Geographic routing on duty-cycled sensors. In *ACM/IEEE IPSN*, pages 440–449, 2007.

[26] J. Benson, T. O'Donovan, U. Roedig, and C.J. Sreenan. Opportunistic aggregation over duty cycled communications in wireless sensor networks. In *ACM/IEEE IPSN*, pages 307–318, 2008.

[27] I. Rhee, A. Warrier, M. Aia, J. Min, and M.L. Sichitiu. Z-MAC: a hybrid MAC for wireless sensor networks. *IEEE/ACM TON*, 16(3):511–524, 2008.

[28] M. Buettner, G.V. Yee, E. Anderson, and R. Han. X-MAC: a short preamble MAC protocol for duty-cycled wireless sensor networks. In *Proceedings of ACM Sensys*, page 320, 2006.

[29] Y. Sun, O. Gurewitz, and D.B. Johnson. RI-MAC: a receiver-initiated asynchronous duty cycle MAC protocol for dynamic traffic loads in wireless sensor networks. In *ACM Sensys*, pages 1–14, 2008.

[30] R. Zheng, J.C. Hou, and L. Sha. Asynchronous wakeup for ad hoc networks. In *Proceedings of ACM MobiHoc*, pages 35–45. ACM, 2003.

[31] P. Dutta, S. Dawson-Haggerty, Y. Chen, C.J.M. Liang, and A. Terzis. Design and evaluation of a versatile and efficient receiver-initiated link layer for low-power wireless. In *ACM Sensys*, pages 1–14, 2010.

[32] B. Raman, K. Chebrolu, S. Bijwe, and V. Gabale. PIP: a connection-oriented, multi-hop, multi-channel TDMA-based MAC for high throughput bulk transfer. In *ACM Sensys*, pages 15–28, 2010.