A^2-MAC: An Adaptive, Anycast MAC Protocol for Wireless Sensor Networks

Hwee-Xian Tan and Mun Choon Chan
Department of Computer Science, School of Computing, National University of Singapore
{hweexian, chanmc}@comp.nus.edu.sg

***Abstract*—Energy constraints in wireless sensor nodes necessitate the design and development of energy-efficient MAC protocols to arbitrate access to the shared communication medium. While there exists a plethora of sensor MAC protocols, these protocols do not individually vary the duty-cycle of each sensor according to local connectivity status, to maximize energy savings.**

In this paper, we propose *A^2-MAC* - an *A*daptive, *A*nycast *MAC* protocol for low-powered wireless sensor networks. It utilizes: (i) a random wakeup schedule, such that each node can independently and randomly wakeup in each cycle without coordination and time synchronization; (ii) adaptive duty-cycles based on network topology; and (iii) adaptive anycast forwarders selection, which allows each node to transmit to any member in its forwarding set. There are two key adaptive mechanisms in A^2-MAC: (i) each node varies its duty-cycle and set of forwarding nodes such that energy consumption can be locally minimized for a given local delay performance objective; (ii) nodes cooperatively reduce the duty-cycles required of their forwarding nodes, depending on local network connectivity. By allowing nodes to operate with different duty-cycles and forwarding sets, A^2-MAC achieves better energy-latency tradeoffs and extends node lifetime substantially, while providing good end-to-end latency.

I. Introduction

Advancements in wireless networking and technology have led to the proliferation of tiny computing and sensing devices that are capable of performing collaborative tasks such as tactical surveillance and environmental monitoring. These network elements communicate with one another via multi-hops without centralized control and are densely deployed to maximize sensing coverage. However, the inherent nature of wireless sensor networks, such as intermittent connectivity and energy constraints, poses challenges to their deployment and operation. In particular, the severe energy limitation of sensor nodes has received much focus in the research community.

To reduce energy consumption and prolong network lifetime, sensor networks are usually duty-cycled. Each node remains in low-power sleep mode most of the time, and wakes up periodically to sense for channel activities. The Medium Access Control (MAC) layer is responsible for arbitrating access to the shared medium in a fair and efficient manner. Typically, sensor MAC protocols incorporate wakeup schedules into the medium access control operation, such that nodes need not monitor the channel continuously for communication.

Performance studies [1][2][3][4][5][6][7][8] show that while wakeup schedules are effective in reducing energy consumption in sensor networks due to the sporadic characteristics of sensor traffic, the delay incurred by waiting for the next-hop forwarding node to be awake, viz. *sleep latency*, can be quite large. For example, a 1% duty-cycle can potentially reduce the energy consumption of a network by 99% when no traffic is being generated. However, the expected per-hop sleep latency of a packet is 50% of the cycle period.

The wakeup schedule is a key component in the design of a duty-cycled MAC to reduce energy consumption. Synchronous schemes such as S-MAC [2][3], T-MAC [4], D-MAC [5] and R-MAC [6] require nodes to synchronize with each other, which can be complex and expensive especially in large multi-hop networks with clock drifts, low duty-cycles and transient link qualities. Reduction in sleep latency is thus achieved at the expense of substantial control overhead. Asynchronous schemes such as B-MAC [7], X-MAC [1] and C-MAC [8] rely on preambles to coordinate access to the channel and do not require synchronization. While such asynchronous schemes are energy-efficient, they still spend significant energy on idle listening and preamble sampling. The end-to-end latency can also be large (long sleep latency).

In this paper, we propose *A^2-MAC*, an asynchronous *A*daptive, *A*nycast MAC protocol for low-powered sensor networks. It utilizes (i) a *random wakeup schedule*, such that each node can independently and randomly select its wakeup schedule without coordination and time synchronization; (ii) *adaptive duty-cycles* based on network topology; and (iii) *adaptive anycast forwarders* selection, which allows each node to transmit to any member to its forwarding set and effectively reducing expected sleep latency. There are two key adaptations in A^2-MAC: (i) each node adaptively varies its duty-cycle and set of forwarding nodes such that energy consumption is locally minimized for a delay performance objective; and (ii) nodes cooperatively reduce the duty-cycles required of their forwarding nodes, depending on local network connectivity.

By exploiting the redundancy of dense network deployments as well as combining random schedules and anycast mechanisms, nodes can operate with different duty-cycles and forwarding sets to reduce energy consumption. We compare A^2-MAC with X-MAC and the optimal protocol in [9] (hereafter referred to as opt-MAC for brevity) whereby all nodes use the same duty-cycle. Our evaluation shows that A^2-MAC achieves better energy-latency trade-offs and extends node lifetime substantially while providing good end-to-end latency.

The rest of this paper is organized as follows. Section II discusses related work. Sections III and IV detail the basic and adaptive (forwarder and duty-cycle selection) components of A^2-MAC respectively. We evaluate the performance of A^2-MAC in Section V and conclude in Section VI.

II. Related Work

In the pioneer work on sensor MAC protocols, Ye et al [2] identifies the main sources of energy consumption in any contention-based MAC protocol as: (i) collision; (ii) overhearing; (iii) control packet overhead; and (iv) idle listening. It is highlighted that in sensor networks without energy awareness, nodes expend most of their energy in idle listening due to the sporadic nature of data traffic. Consequently, subsequent works on sensor MAC protocols always incorporate some form of wakeup scheduling such that nodes do not remain awake throughout the entire network lifetime but wakeup at intervals for communication and to check for channel activity.

Wakeup mechanisms can be broadly classified as: (i) on-demand; (ii) synchronous; and (iii) asynchronous. Sensor MAC protocols that make use of on-demand wakeup mechanisms [10] require out-of-band signaling (using a low power radio) in order to wake up the nodes in time for data reception.

In synchronous wakeup (or scheduled rendezvous) schemes such as S-MAC [2][3], T-MAC [4], D-MAC [5] and R-MAC [6], nodes wakeup during the same designated time slots to communicate, thus effectively reducing idle listening. However, tight time synchronization and pre-negotiation of schedules are necessary, which incurs high overheads.

In asynchronous wakeup schemes such as B-MAC [7] and X-MAC [1], the schedules of the sender(s) and the receiver(s) are decoupled, thereby removing the need for any synchronization. Nodes wake up periodically to check for any channel activity - a technique commonly known as LPL (Low Power Listening). If channel activity is detected, the node remains awake to receive the incoming packet; otherwise, the node resumes sleeping. These asynchronous MAC protocols are unicast in nature and use the same duty-cycle for each node.

Anycast MAC schemes [8][9], whereby a transmitting node sends a packet to any one of the members within a forwarding set, have been proposed in the context of sensor networks. A key difference between A^2-MAC and many existing anycast protocols is that the latter uses the same duty-cycle for all nodes, while A^2-MAC varies duty-cycles on a per-node basis, leading to better energy-latency trade-offs. Finally, [11] studies the impact of unreliable communication links on data forwarding in duty-cycled networks. One interesting result as claimed by the authors is that opportunistic looping can potentially reduce the overall delay. However, ways to design the wakeup schedule or adapt the duty-cycle are not presented, but assumed to be given inputs to the problem.

III. Protocol Details

A. Basic Mechanism

The wakeup schedule of A^2-MAC is based on an asynchronous slot model. The schedule in each cycle is divided into: (i) active (listening) slots, in which nodes wakeup and monitor the channel for activities (analogous to LPL in asynchronous MAC schemes); and (ii) inactive (sleep) slots, in which nodes switch to low-powered sleep mode by default.

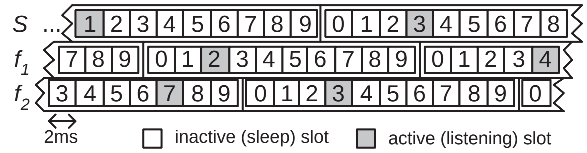


Fig. 1. (10, 1, 2ms) random wakeup function with 3 unsynchronized nodes.

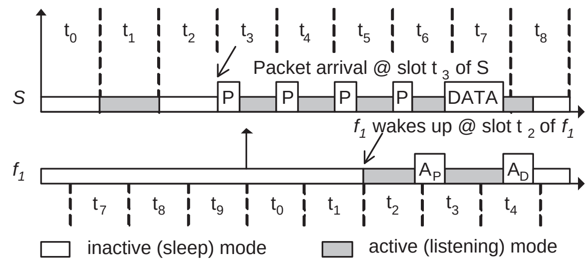


Fig. 2. Data transfer in A^2-MAC between 2 unsynchronized nodes.

We consider a random wakeup scheduling function represented as a (v, α, τ) design, where v is the total number of slots per cycle (chosen to achieve a sleep latency constraint); $\alpha \leq v$ is the number of *active* listening slots per cycle; and τ is the duration of each slot in the duty-cycle such that cycle length is $v.\tau$. At the beginning of each cycle, each node selects α out of v slots to be active in, such that its active/awake probability in any slot is $\frac{\alpha}{v}$. This selection is done randomly and independently of other nodes to eliminate coordination and synchronization overheads, as well as provide ease of adaptation of duty-cycles. In the (10, 1, 2ms) random wakeup function of Figure 1, nodes S, f_1 and f_2 each selects 1 active slot out of 10 available slots within a cycle, and the slots of each node may be unsynchronized with one another.

Compared to synchronous schedules, the number of *active* one-hop neighbors during an arbitrary time slot in A^2-MAC is reduced, which effectively minimizes collisions and reduces overhearing. However, the default active slots of each node are unlikely to overlap, particularly when duty-cycle is low. Hence, A^2-MAC uses a probing mechanism to guarantee communication between the transmitter and its forwarder (if one exists) within a single cycle period. In Figure 2, S wakes up at (its) slot t_1 according to its wakeup function in Figure 1 and resumes sleeping at t_2. Upon a packet arrival at t_3, S switches to active (listening) mode and continuously probes its neighbors using short preambles P in every subsequent slot. Probing terminates when S receives a preamble acknowledgement A_P at t_6 from a forwarder f_1 that is awake. Transmission completes when S transmits data to f_1 during t_7 and receives a corresponding data acknowledgement A_D in t_8.

The probing for active neighbors does not incur additional delays or overheads as compared to existing asynchronous MAC protocols, as all such protocols *have* to transmit preambles up to duration of a cycle period to guarantee commu-

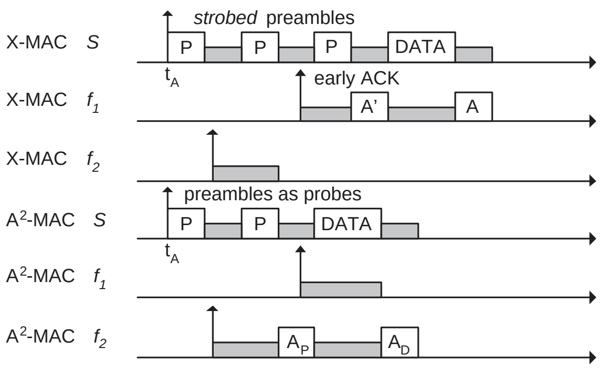


Fig. 3. Protocol details of X-MAC and A^2-MAC.

nication between nodes. Figure 3 illustrates behaviors of X-MAC and A^2-MAC upon a packet arrival at time t_A at S. Due to the unicast nature of X-MAC, f_2 cannot forward packets for S even though it wakes up before f_1, as f_1 is the designated forwarder for S. In contrast, A^2-MAC achieves shorter delays and incurs less overheads using anycast. When collisions of A_Ps occur due to the presence of multiple awake forwarders that detect P and transmit their A_Ps at the same time, each forwarder backoffs for a randomly chosen interval before attempting to retransmit its A_P. When there is only one forwarding node, A^2-MAC behaves similarly to X-MAC.

B. Advantages of Combining Anycast with Random Schedules

Besides path diversity and multipath routing [8][12][13], the anycast mechanism can provide robustness to intermittent link connectivity and latency reduction in a duty-cycled MAC.

1) Robustness to Intermittent Link Connectivity: The transient characteristics of the physical layer leads to intermittent link connectivity. Typical MAC protocols attempt multiple retransmissions across the same temporally-broken link before a link failure is ascertained and an alternative routing path is utilized. With anycast, a node can dynamically select its forwarder based on prevailing link conditions and provide robustness to intermittent connectivity.

2) Reduction in Latency: In contrast to synchronized schemes where active slots are congregated together, A^2-MAC enables packets to be transmitted across multiple hops in one cycle. Considering the (10, 1, 2ms)-design in Figure 1, a packet can be transmitted from S to f_1 in slots t_5 and t_6 (of S), and f_1 to f_2 in slots t_6 and t_7 (of f_1). As the active slots in a cycle are randomly chosen, the average sleep latency T_i (measured in slots) of node i before any one of its forwarders is awake is dependent on v and α, computed by:

$$T_i = \frac{v}{\sum_{j \in F_i} \alpha_{ij} + 1}, \tag{1}$$

where F_i is the forwarding set of i; and α_{ij} is the awake probability (duty-cycle) required of forwarder $j \in F_i$ by i.

C. Interaction with Routing Protocol

A^2-MAC is inter-operable with any routing protocol that provides: (i) a set of candidate forwarding nodes; and (2) a metric that indicates the *progress* made by each forwarder. Examples of such metrics include hopcount to destination, geographical distance [13] and ETX [14]. In this paper, we use the Maximum Forward Progress (MFP) [15] routing metric, which forwards packets based on geographical locations. Only neighbors with positive progress (closer to the sink than the transmitter) are considered eligible forwarders. As no state information is required in MFP, this allows us to study the performance of A^2-MAC without routing overheads.

IV. Adaptation in A^2-MAC

The primary objective of A^2-MAC is to reduce the duty-cycles (and energy consumption) of nodes, in order to extend network connectivity and coverage, and subject to a desired delay constraint $d_{\max}$. In this section, we describe the two key adaptation components of A^2-MAC, viz. forwarder selection and duty-cycle selection, that help to achieve this objective.

The **candidate set** $\aleph_i$ of an arbitrary node i is the set of one-hop neighbors with *positive* progress towards the destination (sink). $\aleph_i$ can be learnt via a simple neighbor discovery scheme during network initialization. For each candidate node $j \in \aleph_i$, d_{ij} denotes the progress made by i when it transmits its packet to the sink via j and $\alpha_{ij} \in (0, v]$ denotes the duty-cycle required of j by i (in a cycle with v slots). The **forwarding set** $F_i \subseteq \aleph_i$ is the set of neighbors within the candidate set that are selected to forward packets from i to the sink.

We define $D_i = \frac{\sum_{j \in F_i} (d_{ij}\alpha_{ij})}{\sum_{j \in F_i} \alpha_{ij}}$ to be the average per-hop progress. The corresponding average per-hop rate of progress is given by $V_i = \frac{D_i}{T_i}$, where T_i is the sleep latency in Equation 1. Generally, the inclusion of more forwarders decreases T_i; however, inclusion of forwarders with small progress (d_{ij}) decreases the average progress D_i and rate of progress V_i.

The minimum average per-hop rate of progress required to satisfy the delay constraint $d_{\max}$ is $V_{min} = \frac{D_{\max}}{d_{\max}}$, where $D_{\max}$ is the maximum 'distance' from any node to the sink. Then, the selection process in node i has to find the set of forwarders F_i and their associated duty-cycles α_{ij} $\forall j \in F_i$ such that: (i) $V_i \geq V_{min}$ to meet the rate of progress and delay constraints; (ii) maximum duty-cycle required of each forwarder ($\max \alpha_{ij}$) is minimized, to prolong network connectivity and coverage.

A. Forwarding Set and Duty-Cycle Selection

We first present two lemmas that are useful in the selection process for the forwarding set and duty-cycle of each node.

Lemma 1: Let the set of candidate nodes $\aleph_i$ of node i be sorted in descending order of progress, from 1 to $|\aleph_i|$. The optimal set of forwarders $F_{opt(i)} \subseteq F_i$ that minimizes $\max_{j \in \aleph_i} \alpha_{ij}$ is the first n_i forwarders with the largest progress.

Lemma 2: To meet the rate of progress constraint V_{min}, $\max_{j \in F_i} \alpha_{ij}$ is minimized among all forwarders iff their associated duty-cycles are the same, i.e. $\alpha_{ij} = \alpha_{ik}$ $\forall j, k \in F_i$.

The proofs for these two lemmas can be found in [16], and they provide guidelines on how the forwarding set and duty-cycle of each node should be selected to achieve the

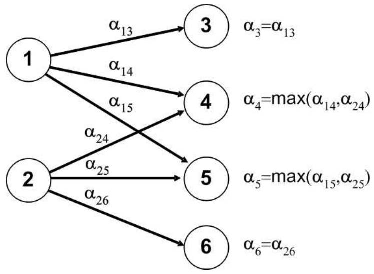


Fig. 4. Computation of forwarding sets and duty-cycles.

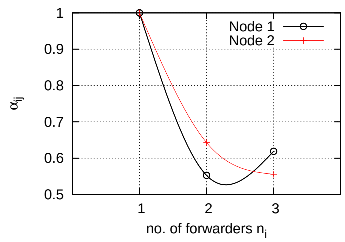


Fig. 5. α_{ij} versus n_i for two different nodes.

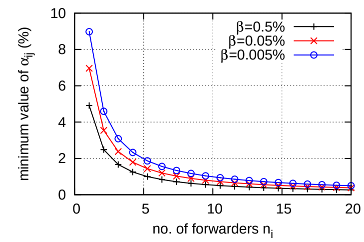


Fig. 6. Minimum α_{ij} required for varying n_i.

objectives of A^2-MAC, i.e. to reduce duty-cycles, as well as extend connectivity and coverage, subject to a delay constraint.

Initially, all the nodes in the candidate set $\aleph_i$ are sorted in descending order of progress. Each candidate node is then incrementally added into the forwarding set F_i. We use $F_i^{(\varphi)}$ to denote the forwarding set containing the first φ nodes in $\aleph_i$ with the most progress, where $1 \leq \varphi \leq |\aleph_i|$. For each forwarding set $F_i^{(\varphi)} \subseteq \aleph_i$, we compute the minimum α_{ij} needed to ensure that $V_i \geq V_{min}$. The forwarding set with the smallest α_{ij} is considered to be optimal for node i.

We illustrate the selection process using the example in Figure 4 with $V_{min} = 2$ and $v = 1$. Node 1 has three candidate nodes such that $\aleph_1 = \{3, 4, 5\}$, with corresponding progresses $d_{13} = 1$, $d_{14} = 0.9$ and $d_{15} = 0.2$. The minimum duty-cycles required for the three forwarding set combinations are $\alpha_{13} = 1$, $\alpha_{13} = \alpha_{14} = 0.5526$ and $\alpha_{13} = \alpha_{14} = \alpha_{15} = 0.619$ using the forwarding sets {N3}, {N3, N4} and {N3, N4, N5} respectively, as illustrated in Figure 5. Hence, the minimum duty-cycle is obtained when only {N3, N4} are used as forwarders, resulting in $\alpha_{13} = \alpha_{14} = 0.5526$.

Similarly, node 2 has three candidate nodes such that $\aleph_2 = \{4, 5, 6\}$, with progresses $d_{24} = 1$, $d_{25} = 0.75$ and $d_{26} = 0.5$. The duty-cycles required are $\alpha_{24} = 1$, $\alpha_{24} = \alpha_{25} = 0.6429$ and $\alpha_{24} = \alpha_{25} = \alpha_{26} = 0.5556$ respectively for the forwarding sets {N4}, {N4, N5} and {N4, N5, N6}. In this case, all three forwarders should be used to obtain the minimum duty-cycle. As the duty-cycles required of forwarder 4 by nodes 1 and 2 are different, the final duty-cycle of node 4 is given by $\alpha_4 = \max\{\alpha_{14}, \alpha_{24}\} = 0.5556$.

We note that when $\sum_{j \in F_i} \alpha_{ij}$ is small, the maximum sleep latency T_i for node i can be large. T_i can be bounded by ensuring that the probability of having *no* forwarders active throughout a cycle with v time slots is less than a specified QoS threshold β ($0 < \beta < 1$), such that:

$$P_n = [\prod_{j \in F_i} (1 - \frac{\alpha_{ij}}{v})]^v \leq \beta. \quad (2)$$

Figure 6 illustrates the minimum average α_{ij} (as a percentage of v) required using varying values of n_i.

B. The Adaptation Algorithm

Adaptation is performed during network initialization and topological changes, which allows each node i to compute: (i) its forwarding set F_i; (ii) duty-cycles α_{ij} required **of** each forwarder $j \in F_i$; and (iii) its own duty-cycle α_i based on the requirements **from** its neighbors. Initially, the duty-cycles of all nodes are considered to be **undetermined**; for brevity, we refer to such nodes as 'undetermined nodes'.

Algorithm 1 Computing F_i and α_{ij} by i in each round.

1: Input: set of undetermined candidates $\aleph_i^u$, set of determined candidates $\aleph_i^d$, set of candidates $\aleph_i = \aleph_i^u \bigcup \aleph_i^d$, progress $d_{ij}\ \forall j \in \aleph_i$, duty-cycle $\alpha_{ij}\ \forall j \in \aleph_i^d$
2: per-hop rate of progress $V_i = 0$; forwarding set $F_i = \aleph_i^d$; temporary set of undetermined candidates $Q_i^u = \aleph_i^u$; duty-cycles of undetermined candidates $\alpha_{ij} = 0\ \forall j \in \aleph_i^u$
3: $\varphi = |F_i|$; current forwarding set $F_i^{(\varphi)} = F_i$, which contains the $|\aleph_i^d|$ determined candidates and next $\varphi - |\aleph_i^u|$ candidates with largest progress
4: **while** $Q_i^u \neq \emptyset$ and $[(V_i < V_{min}) \ || \ (P_n > \beta)]$ **do**
5:     Compute V_i, $\min \alpha_{ij}^{(\varphi)}$ and P_n using $F_i^{(\varphi)}$
6:     undetermined candidate with next largest progress $b = \underset{j}{\text{argmax}}\ d_{ij},\ j \in Q_i^u$
7:     $\varphi = \varphi + 1$; $F_i^{(\varphi)} = F_i^{(\varphi)} \bigcup \{b\}$; $Q_i^u = Q_i^u \setminus \{b\}$
8: **end while**
9: $\phi = \underset{\varphi}{\text{argmin}}\ \alpha_{ij}^{(\varphi)}$; $\alpha_{ij} = \alpha_{ij}^{(\phi)}\ \forall j \in \aleph_i^u$; $F_i = F_i^{(\phi)}$

Each execution of the adaptation algorithm proceeds in bi-phase rounds. Algorithm 1 summarizes how, in the first phase of every round, a node i with undetermined candidate nodes computes: (i) its forwarding set; and (ii) duty-cycle requirements **of** each forwarder j. These computations are based on the two lemmas presented in Section IV-A. In each iteration of the while loop, the undetermined candidate node that has the largest progress is added to the (current) forwarding set and the new duty-cycle is computed. The loop exits when the local constraints are met (Line 4). A key feature of A^2-MAC is that it exploits higher duty-cycles of determined nodes to reduce the required duty-cycles of additional (undetermined) nodes[1]. The final forwarding set F_i and duty-cycle α_{ij} that is required from its neighbors in the current round is the configuration that provides the minimum duty-cycle requirements.

In the second phase of every round, each undetermined node i computes its *interim* duty-cycle based on the duty-cycle re-

[1]Considering the network topology in Figure 4, once the larger α_4 value of 0.5556 is selected, α_3 can be reduced slightly from 0.5526 to 0.551.

TABLE I
SIMULATION PARAMETERS

Parameter	Value
Transmitting I_{t_x}	11.0 mA
Receiving I_{r_x}	19.7 mA
Idle I_{idle}	0.426 mA
Sleep I_{sleep}	0.001 mA
A^2-MAC Time Slot Length τ	20 ms
A^2-MAC Cycle Length L_{cycle}	2 s

quirements **from** its neighbors (computed from the first phase). The undetermined node with the largest interim duty-cycle among all its undetermined neighbors then fixes its duty-cycle to be that of the computed interim, and thereafter is known as a '**determined** node'. The next round then commences, until all the nodes in the network become determined. The algorithm is guaranteed to terminate, as at least one undetermined node becomes determined in each round.

V. SIMULATION EVALUATION

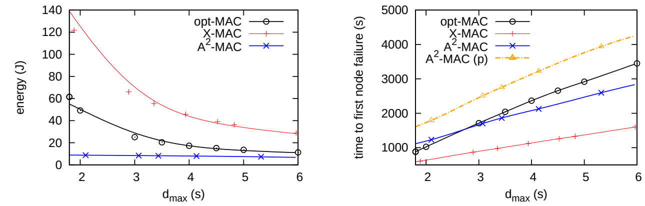
We evaluate the performance of A^2-MAC with: (i) X-MAC [1], a well-known energy-efficient asynchronous unicast MAC protocol; and (ii) opt-MAC [9], which is optimal among approaches using the *same* duty-cycle for all the nodes, using GloMoSim [17]. Results shown are the average of 20 runs. The common simulation parameters are listed in Table I [2]. All traffic arrivals follow a Poisson distribution. The routing protocol used to forward each 60-byte packet is MFP (Maximum Forward Progress). The sink is placed at the top right-hand corner of the terrain of size 250 m × 250 m, and the transmission range is approximately 60 meters. Sections V-A to V-C assume that energy expended in packet transmission is negligible as compared to energy expended through long periods of idle listening. In Section V-D, we consider higher traffic loads where transmissions incur significant energy.

A. Delay Tradeoffs

We vary the delay constraint $d_{\max}$ from 2s to 6s and study the tradeoffs of the three protocols (A^2-MAC, X-MAX and opt-MAC) in a network of 150 randomly placed nodes with average node degree of $d_v \approx 20$ in Figure 7. As $d_{\max}$ increases, nodes sleep longer, leading to lower duty-cycles and per-node energy consumption in Figure 7(a). A^2-MAC achieves better energy-delay tradeoffs particularly for smaller values of $d_{\max}$, due to its complementary use of random wakeup schedules, anycast, as well as adaptive mechanisms. In contrast, X-MAC and opt-MAC assign the same (maximum) duty-cycle to all the nodes, which increases energy consumption.

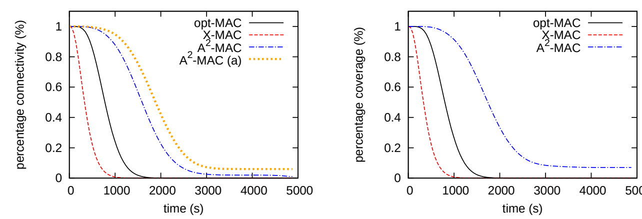
As A^2-MAC does not *globally* optimize the time to the first node failure, it performs slightly worse than opt-MAC (which is optimized for this aspect) for higher $d_{\max}$ values in Figure 7(b). In opt-MAC and X-MAC, nodes are assigned the same duty-cycles and fail at the same rate; in A^2-MAC, nodes

[2]Based on Chipcon CC2420 RF Transceiver specifications.



(a) Energy (b) Time to First Node Failure

Fig. 7. Delay tradeoff under varying delay constraints.



(a) Percentage Connectivity. (b) Percentage Coverage.

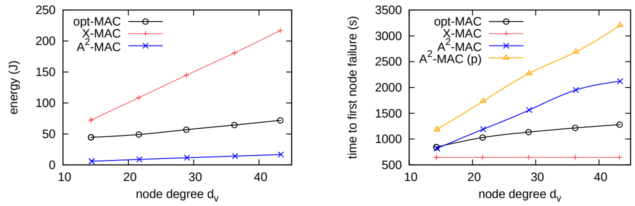
Fig. 8. Percentage connectivity and coverage with $d_{\max} = 2s$.

fail gracefully over time. Consequently, the time to network partition for A^2-MAC - denoted as A^2-MAC(p) - exceeds the time to first node failure of opt-MAC by 20% to 50%, as the network remains connected even when some nodes in the (typically dense) sensor network has failed.

B. Connectivity and Coverage

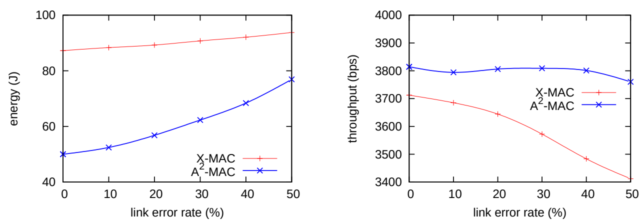
Figure 8 illustrates the network connectivity and coverage over time with $d_{\max} = 2s$. The percentage connectivity is the ratio of nodes that remain alive and connected to the sink relative to total number of nodes. The percentage coverage is the ratio of the terrain within the range of any connected and alive node relative to the initial coverage area.

In X-MAC, nodes do not exploit the redundancy of neighbors to reduce duty-cycles; hence the percentage connectivity deteriorates very quickly over time. Although opt-MAC utilizes anycast to minimize duty-cycles, its network connectivity still deteriorates quickly as all nodes use the same maximum required duty-cycle. Figure 8(a) shows that A^2-MAC has the best percentage connectivity as it: (i) minimizes the local maximum duty-cycle; and (ii) adaptively assigns (different) duty-cycles to each node based on its local topology. We note that the percentage of *alive* nodes in A^2-MAC - denoted as A^2-MAC(a) - is higher than the percentage connectivity; this indicates that there are nodes that are alive but have lost connectivity to the sink. They are potentially useful as they can transmit data to the sink when the network is repaired, or through techniques such as message ferrying. The higher network connectivity in A^2-MAC allows it to achieve better coverage than opt-MAC and X-MAC in Figure 8(b). Notice that in A^2-MAC, a small percentage of the nodes remain connected and cover a small proportion of the network for a long time. These nodes are close to the sink and have few upstream nodes, resulting in extremely low energy consumption.



(a) Energy consumption per node. (b) Time to first node failure.

Fig. 9. Performance with varying network densities and $d_{\max} = 2$.



(a) Energy Consumption (b) Throughput

Fig. 10. Performance with intermittent link connectivity.

C. Random Topology with Varying Network Densities

The network size in Figure 9 is varied from 100 to 300 nodes such that d_v varies between 15 to 45. Figure 9(a) indicates that energy expended increases with increasing d_v. As X-MAC employs a unicast mechanism, it does not exploit the availability of increased redundancy (or neighbors), resulting in high duty-cycles and energy consumption. By utilizing larger forwarding sets as d_v increases, opt-MAC and A^2-MAC can achieve low energy consumption. The latter expends the least energy as it selects neighbors that provide more progress as forwarders, and allows nodes to adapt (lower) their duty-cycles according to local topologies. In Figure 9(b), A^2-MAC and opt-MAC utilize more forwarders with increasing d_v, resulting in lowered duty-cycles and longer times to first node failure. A^2-MAC has better performance and longer time to network partition as its anycast and adaptive mechanisms maximize the benefit of network redundancy.

D. Random Topology with Intermittent Link Connectivity

In Figure 10, 50% of the 100 nodes in the network are randomly selected as data sources and 10% of the links are randomly selected to have error rates from 0% to 50%. $d_{\max}$ is set to 300ms when there is no link error. As opt-MAC utilizes a similar anycast approach as A^2-MAC, its performance in intermittently connected networks is similar to A^2-MAC and not shown. Generally, the number of retransmissions required increases with link errors. This results in the corresponding increase in energy consumption in Figure 10(a). During packet losses, X-MAC retransmits unsuccessfully over the same poor-quality link, resulting in high energy consumption and low throughput. By dynamically selecting the next-hops based on prevailing network conditions, A^2-MAC is more resilient to intermittent link failures; hence it can achieve higher and more consistent throughput in Figure 10(b).

We have also evaluated the performance of the MAC protocols under varying traffic loads and delay constraints d_{max}. The results show that the anycast and adaptive mechanisms in A^2-MAC minimizes excessive overheads and energy consumptions, without compromising on the end-to-end latency.

VI. Conclusion

Severe energy limitations in sensors accentuate the need for energy-efficient MAC protocols. However, duty-cycling incurs higher latencies as transmitters have to wait for forwarders to be awake before communication can commence. A^2-MAC is an adaptive, anycast-based MAC protocol that utilizes an asynchronous random wakeup schedule, anycast mechanism as well as adaptive forwarding set selection and duty-cycle selection. It adapts its duty-cycle and forwarding set based on local network topology and a given delay constraint to achieve energy efficiency with low latencies. A^2-MAC can also achieve better connectivity and coverage, and significantly outperforms existing asynchronous sensor MAC protocols.

References

[1] M. Buettner, G. Yee, E. Anderson, and R. Han, “X-MAC: A Short Preamble MAC Protocol For Duty-Cycled Wireless Networks,” in *SenSys*, 2006.

[2] W. Ye, J. Heidemann, and D. Estrin, “An Energy-Efficient MAC Protocol for Wireless Sensor Networks,” in *INFOCOM*, 2002.

[3] ——, “Medium Access Control with Coordinated Adaptive Sleeping for Wireless Sensor Networks,” *IEEE ACM Transactions on Networking*, vol. 12, no. 3, 2004.

[4] T. van Dam and K. Langendoen, “An Adaptive Energy-Efficient MAC Protocol for Wireless Sensor Networks,” in *SenSys*, 2003.

[5] G. Lu, B. Krishnamachari, and C. Raghavendra, “An Adaptive Energy-Efficient and Low-Latency MAC for Data Gathering in Sensor Networks,” in *International Workshop on Algorithms for Wireless, Mobile, Ad Hoc and Sensor Networks (WMAN)*, 2004.

[6] S. Du, A. K. Saha, and D. B. Johnson, “RMAC: A Routing-Enhanced Duty-Cycle MAC Protocol for Wireless Sensor Networks,” in *INFOCOM*, 2007.

[7] J. Polastre, J. Hill, and D. E. Culler, “Versatile Low Power Media Access for Wireless Sensor Networks,” in *SenSys*, 2004.

[8] S. Liu, K. W. Fan, and P. Sinha, “CMAC: An Energy Efficient MAC Layer Protocol Using Convergent Packet Forwarding for Wireless Sensor Networks,” in *SECON*, 2007.

[9] J. Kim, X. Lin, N. Shroff, and P. Sinha, “On maximizing the lifetime of delay-sensitive wireless sensor networks with anycast,” in *INFOCOM*, 2008.

[10] E. Shih, P. Bahl, and M. J. Sinclair, “Wake on Wireless: an Event Driven Energy Saving Strategy for Battery Operated Devices,” in *MobiCom*, 2002.

[11] Y. Gu and T. He, “Data forwarding in exremely low duty-cycle sensor networks with unreliable communication links,” in *Sensys*, 2007.

[12] S. Jain and S. R. Das, “Exploiting Path Diversity in the Link Layer in Wireless Ad Hoc Networks,” in *WOWMOM*, 2005.

[13] M. Zorzi and R. R. Rao, “Geographic random forwarding (GeRaF) for ad hoc and sensor networks: multihop performance,” *IEEE Transactions on Mobile Computing*, vol. 2, no. 4, 2003.

[14] D. Couto, D. Aguayo, J. Bicket, and R. Morris, “A high throughput path metric for multi-hop wireless routing,” in *MOBICOM*, 2003.

[15] W. Ye, J. Heidemann, and D. Estrin, “Performance of Energy-Aware and Link-Adaptive Routing Metrics for Ultra Wideband Sensor Networks,” *Mobile Networks and Applications*, vol. 11, no. 4, 2006.

[16] H. X. Tan and M. C. Chan, “A^2-MAC: An Adaptive, Anycast MAC Protocol for Wireless Sensor Networks,” School of Computing, National University of Singapore, Tech. Rep., 2009, http://www.comp.nus.edu.sg/~tanhweex/research/a2mac_2009.pdf.

[17] X. Zeng, R. Bagrodia, and M. Gerla, “GloMoSim: A Library for Parallel Simulation of Large-Scale Wireless Networks,” in 12th *Workshop on Parallel and Distributed Simulation*, 1998.