Predictive Duty Cycle Adaptation for Wireless Camera Networks

Paul J. Shin, Henry Medeiros, Johnny Park, and Avinash Kak
School of Electrical and Computer Engineering
Purdue University, West Lafayette, Indiana 47907-2035
{paulshin, hmedeiro, jpark, kak}@purdue.edu

Abstract*—Wireless sensor networks (WSN) typically employ dynamic duty cycle schemes to efficiently handle different patterns of communication traffic in the network. However, existing duty cycling approaches are not suitable for event-driven WSN, in particular, camera-based networks designed to track humans and objects. A characteristic feature of such networks is the spatially-correlated bursty traffic that occurs in the vicinity of potentially highly mobile objects. In this paper, we propose a concept of indirect sensing in the MAC layer of a wireless camera network and an active duty cycle adaptation scheme based on Kalman filter that continuously predicts and updates the location of the object that triggers bursty communication traffic in the network. This prediction allows the camera nodes to alter their communication protocol parameters *prior* to the actual increase in the communication traffic. Our simulations demonstrate that our active adaptation strategy outperforms TMAC not only in terms of energy efficiency and communication latency, but also in terms of TIBPEA, a QoS metric for event-driven WSN.

I. Introduction

In designing a wireless sensor network, one attempts to maximize both the lifetime of the nodes and the network performance. Duty cycling the radio at the nodes is considered to be one of the most effective ways to conserve energy, and obviously such energy conservation directly increases the lifetime of the network. However, changing the duty cycle also affects the communication latency at the nodes. As a result, trying to minimize both the latency and the energy expenditure involves a fundamental tradeoff.

Various approaches have been proposed to improve this tradeoff. In addition to a class of static duty cycling schemes [1], [2], [3], [4], [5], there are several approaches that employ adaptive or dynamic duty cycling mechanisms, such as TMAC [6], AMAC [7], DSMAC [8], and CMAC [9]. In such schemes, the dynamic duty cycle adaptation of a node is predicated by the detection of changes in the *current* traffic conditions at the node. Although there exist event-driven MAC protocols that try to minimize the latency of either a subset or all of the event-triggered messages by removing redundancy among packets [10] or by exploiting multiple channels [11], these efforts are still made after an event of interest actually occurs.

Passive duty cycle adaptation schemes mentioned above are not suitable for event-driven WSNs, such as wireless camera networks (WCNs) intended for tracking humans and objects in motion. In such networks, the events spawned by moving targets continuously trigger the initiation of new communication links between new pairs of nodes and new routing paths. The passive schemes for duty cycle adaptation perform poorly under such conditions due to the inherent delay between the detection of a new event and the reaction to the event in terms of communication. Those methods work best when the same communication links or the same routing paths are reused repeatedly even in the vicinity of an event.

In this paper, we propose the predictive duty cycle adaptation (PDCA) scheme specifically designed for event-driven WSNs. Whereas the existing approaches *passively* adapt the duty cycle according to the current network conditions, the proposed PDCA scheme *actively* adapts the duty cycle in a predictive fashion according to the probability that an event of interest will occur within a node's sensing field in the near future. This probability will be referred to as the *future event detection probability* (FEDP) at a node and computed based on the *spatio-temporal event probability* (STEP) predicted by using a Kalman filter in the MAC layer. To enable the predictive adaptation, each node marks outgoing packets if it detects an event. Neighboring nodes receiving/overhearing these packets can be briefly informed about the event by looking at their MAC header, resulting in the *indirect sensing* from an *augmented sensing region*, which will be explained later.

In the rest of the paper, we first briefly survey some unique features of event-driven WSNs in Section 2. Section 3 then presents our predictive approach to the reconfiguration of the MAC protocol parameters in such networks. The performance evaluation of the proposed approach, carried out in a realistic simulation environment in the context of target tracking, is presented in Section 4. We present the results obtained with our approach vis-a-vis TMAC. Section 5 concludes the paper.

II. Event-Driven Wireless Sensor Networks

Event-driven WSNs that can be typified by a WCN differ from the more traditional WSNs in the sense that the events occurring in the environment are likely to cause the radio broadcast traffic to become bursty among the nodes nearby the event. Due to limited computational power and sensing capability, the sensor nodes in an event-driven WSN usually collaborate with one another in order to detect events and to estimate their various attributes. For tasks such as object detection and tracking, an event-driven WSN may involve computations beyond the capabilities of the processor at any single node. Such tasks would require cluster-based distributed implementations of the algorithms [12], [13]. The collaborative processing required by such algorithms is usually carried

978-1-4577-1707-9/11/$26.00 ©2011 IEEE

out with the help of clusters that consist of nodes that can capture some sensory information related to the event. Such collaborative computations typically require intensive message exchanges within a cluster, resulting in highly bursty communications that, unless the communication protocol parameters are changed in a timely manner, may be characterized by frequent packet collisions that may result in waste of energy and failure in transmitting critical data.

The purpose of an event-driven WSN is to detect an event and perform event-specific tasks in a timely manner. Considering a WCN deployed in a large area for surveillance purposes, it is critical that the event information be updated and transmitted to the end user in a timely manner. This obviously requires a high quality of service (QoS) with regard to the communications in the network. A network must also rate high on appropriate QoS metrics from the standpoint of the communications requirement for collaborative problem solving by the nodes in a cluster.

III. Predictive Duty Cycle Adaptation

The proposed approach, which we refer to as *Predictive Duty Cycle Adaptation* (PDCA) scheme, *actively* adjusts the duty cycle of the nodes that are about to experience an event that may provoke high communication traffic. In other words, the PDCA scheme increases the duty cycle at a node if it is likely that the node will soon experience an event and decreases the duty cycle otherwise. This is contrast to existing approaches that adapt the duty cycle of nodes in a *passive* manner by reacting to the current event. PDCA attempts to meet two important yet conflicting objectives—high energy efficiency and low communication latency—by actively adapting the duty cycle according to the probability that a node will experience an event in the near future. We estimate this probability in terms of the *Spatio-Temporal Event Probability* (STEP), which will be described in detail in Section III-B.

An event could be defined in various ways depending on applications. In the context of object detection and tracking—the primary application we are interested in—we can define an *event* as an object of interest. In this sense, therefore, when an event occurs at a node, it implies that an object of interest is present within the sensing field of a node.

A. Indirect Sensing in MAC Layer

When an event occurs at a node, its communication neighbors must be notified so that they can get ready to handle the imminent increase in radio traffic. The occurrence of an event at a node is made known to its communication neighbors by setting a dedicated bit in the MAC header of packets. We define this dedicated bit as the *Explicit Event Notification* (EEN) bit. Since a node experiencing an event is most likely to generate traffic, embedding event information in the outgoing packet header is enough to notify neighboring nodes of the current event without incurring any additional communication traffic. Since MAC layer protocols are not responsible for acquiring direct sensor measurements, the proposed PDCA scheme provides an interface that allows the application layer to notify the MAC layer that the EEN bit of all outgoing packets should be set when an event is within the sensing field of the node.

When a node directly receives or overhears a packet for which the EEN bit is set, this can be construed as the node *indirectly sensing* the event. The receiving node may assume that the event is located somewhere in the sensing field of the transmitting node. This manner of localizing an event can be interpreted as constituting an *augmented sensing field* for the node receiving or overhearing the packets. Therefore, the augmented sensing field of a node is the union of the sensing field of its one-hop neighbors.

In order to carry out indirect sensing, a node should be aware of the sensing fields of its communication neighbors. For that purpose, our system encompasses an initialization stage during which the node receives this information. Since it is beyond the scope of a MAC protocol to compute the sensing parameters of the node, we assume that after an initial sensor localization (or calibration for cameras) procedure, this information is available to the application layer. The application layer then delivers this information to the MAC layer in the form of a 3-tuple $(i, \mathbf{z}, R)$, where i identifies the node (i_{self} is the local node address), $\mathbf{z}$ corresponds to the center of the sensing field of the node, and R is an ellipsoid that approximates its sensing range. When the node receives the initialization messages from its neighbors, then it simply stores them in a list within the MAC layer.

B. Spatio-Temporal Event Probabilities

Given an event j at a time instant t_k, the corresponding spatio-temporal event probability (STEP) distribution at a particular position $\mathbf{u}$, denoted as $S^j_{k+1|k}(\mathbf{u})$, is given by *the probability of the predicted position of the event j at time* $t_{k+1} = t_k + \delta_k$, where δ_k is a constant larger than the time needed to change the duty cycle. That is, let $\mathbf{p}^j_{k+1|k}$ be the predicted position of the event, then the STEP at a position $\mathbf{u}$ at a time instant t_{k+1} is given by $S^j_{k+1|k}(\mathbf{u}) = Pr(\mathbf{p}^j_{k+1|k} = \mathbf{u})$. Note that $S^j_{k+1|k}(\mathbf{u})$ and $\mathbf{p}^j_{k+1|k}$ correspond to the prediction to a future time instant t_{k+1}. The time interval δ_k may be determined based on the next possible time instant that a node can actually adopt a new schedule. The STEP distribution is updated whenever a new measurement is obtained by indirect sensing. In the following sections we will describe how the detection of an event is used to estimate the most likely position of the event and to predict its future position from which the STEP distribution is obtained.

It is reasonable to assume that as an object is being tracked, each node will acquire multiple observations of the object. The goal of the discussion that follows is to show how all the measurements acquired sequentially as the object is being tracked can be used in a recursive framework to predict as to what nodes are likely to see the object next with what probability. There are several recursive estimation methods that could be used for such purpose, such as the Particle filter [14], [15], but we chose to use the Kalman filter [16] because of its low computational requirements.

1) System Model and Kalman Filter Equations: Each node that is currently engaged in observing and tracking the object of interest will create a state vector for the object. When a new object is detected within the augmented sensing field of a node, the state vector of the object is initialized with the initial event observation. Subsequently, the node uses the Kalman filter equations to update the state vector. This updated state vector is then used to make a prediction about where the event will appear next as it moves.

We model the event state as a 4-D vector that consists of the event position (x_k, y_k) at a discrete time instant k and its velocity $(\dot{x}_k, \dot{y}_k)$. That is, the state vector is given by $\mathbf{x}_k = \begin{bmatrix} x_k & y_k & \dot{x}_k & \dot{y}_k \end{bmatrix}^T$.The system dynamics are modeled by

$$\mathbf{x}_{k+1} = \begin{bmatrix} x_k + \delta_k \dot{x}_k + \frac{a_x}{2}\delta_k^2 \\ y_k + \delta_k \dot{y}_k + \frac{a_y}{2}\delta_k^2 \\ \dot{x}_k + a_x \delta_k \\ \dot{y}_k + a_y \delta_k \end{bmatrix},$$

where δ_k is the time elapsed between two observations. That is, if the k^{th} sample was taken at time t_k, the sample $k+1$ is acquired at time $t_{k+1} = t_k + \delta_k$. The event acceleration (a_x, a_y) is modeled as white Gaussian noise with covariance matrix Q_k. Then, the system dynamics can be represented as $\mathbf{x}_{k+1} = F_k \mathbf{x}_k + W_k \mathbf{w}_k$, where $\mathbf{w}_k$ is the process noise vector with covariance matrix Q_k.

The measurements are given by the approximated position of the event. Since a single bit is used to describe the event in the MAC header, we approximate the position of the event as a Gaussian distribution $(\mu(i), \Sigma(i))$, where i indicates the ID of a packet sender with EEN set, $\mu(i)$ the center of the sensing field of Node i, and $\Sigma(i)$ the ellipsoidal approximation of the sensing field. The measurement model can then be described by $\mathbf{z}_{k+1} = H_{k+1}\mathbf{x}_{k+1} + \mathbf{v}_{k+1}$, where $\mathbf{v}_{k+1}$ is the measurement noise, assumed white Gaussian with covariance matrix R_{k+1}. If a measurement is received along with its corresponding time stamp and a time synchronization is maintained among nodes, we can easily obtain a reasonably precise measurement time and accurately compute δ_k [17]. Let $\hat{\mathbf{x}}_{k+1|k}$ and $\hat{\mathbf{x}}_{k|k}$ be the predicted and the previously estimated state vectors, and similarly, $P_{k+1|k}$ and $P_{k|k}$ be the predicted and the previously estimated covariance matrices. Then, the time update equations of the Kalman filter are given by

$$\hat{\mathbf{x}}_{k+1|k} = F_k \hat{\mathbf{x}}_{k|k} \quad (1)$$

$$P_{k+1|k} = F_k P_{k|k} F_k^T + W_k Q_k W_k^T. \quad (2)$$

The measurement update equations for the filter are given by

$$K_{k+1} = P_{k+1|k} H_{k+1}^T (H_{k+1} P_{k+1|k} H_{k+1}^T + R_{k+1})^{-1} \quad (3)$$

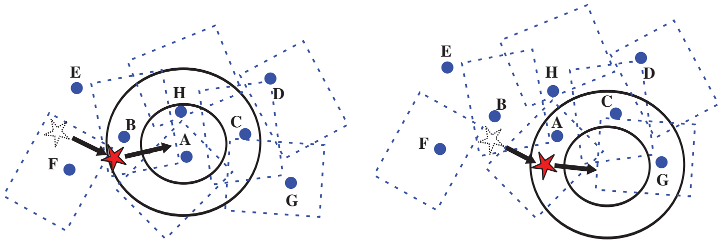
$$\hat{\mathbf{x}}_{k+1|k+1} = \hat{\mathbf{x}}_{k+1|k} + K_{k+1}(\mathbf{z}_{k+1} - H_{k+1}\hat{\mathbf{x}}_{k+1|k}) \quad (4)$$

$$P_{k+1|k+1} = (I - K_{k+1} H_{k+1}) P_{k+1|k}, \quad (5)$$

where K_{k+1} denotes the Kalman gain.

C. Duty Cycle Adaptation

While updating the Kalman filter state as new measurements are available, each node also predicts the probability that an event will occur at a particular position and time instant using the prediction step in the Kalman filter framework. Given this predicted STEP distribution, we first compute how much the STEP overlaps with the sensing field of each node. Next, we show how a node determines its proper duty cycle based on this overlap and how the duty cycle adaptation actually takes place.



(a)

(b)

Figure 1: A depiction of a WCN engaged in tracking a moving object at two subsequent time instants. The dotted rectangles represent the sensing field of nodes. The red star indicates the object of interest and the black solid arrows its moving direction. The regions divided by black solid ellipsoids indicate examples of a contour map of the STEP of the event predicted by a node.

1) Future Event Detection Probability: Given the STEP of an event j at time t_k, each node estimates how likely the event will occur within its sensing field at time t_{k+1}. The probability that an event j will occur within the sensing field of a node i at time t_{k+1}, $S_{k+1|k}^{(j,i)}$, is computed by $S_{k+1|k}^{(j,i)} = \int_{\mathbf{u}\in G(i)} Pr(\mathbf{p}_{k+1|k}^{j} = \mathbf{u}) d\mathbf{u} = \int_{\mathbf{u}\in G(i)} S_{k+1|k}^{j}(\mathbf{u}) d\mathbf{u}$, where $G(i)$ denotes the sensing field of the node i. This probability is computed as the integration of STEP over the sensing field of a node. Since it represents the probability that a node will detect an event in a future time instant, we call it the *Future Event Detection Probability* (FEDP) at a node. The FEDP then plays a role of the metric for determining a proper duty cycle. Computing the exact integration of STEP over the entire sensing field of a node entails a large amount of computation which may not be feasible for real-time operation. Thus, we can use instead an approximation scheme such as the Euclidean distance from the predicted event position to the center of the sensing field or the Mahalanobis distance between them.

2) Determining a Proper Duty cycle: Once an event is detected and its corresponding FEDP value at a node is computed, each node decides its appropriate level of duty cycle: the higher the FEDP, the higher the duty cycle the node adopts. Suppose we have N levels of duty cycle—d_1, d_2, ..., d_N—with d_N being the highest. Let d_c be the current duty cycle level, where $c \in \{1, \ldots, N\}$. Whenever a STEP update occurs, a node will compute its new FEDP and accordingly a new duty cycle level d_m, and schedule a change of duty cycle to be executed at time $t_{k+1} = t_k + \delta_k$. At time t_{k+1}, the node then adopts the new schedule corresponding to the duty cycle level d_m and broadcasts it to its neighbors so that they can be aware of the new communication schedule.

Consider an example illustrated in Figure 1 where an event is being tracked and its future position is predicted. The ellipsoids represent the equiprobable contours of the STEP distribution, which is assumed Gaussian for computational convenience. If the event, which was initially detected by Node

F, moves to the sensing field of Node B thereby triggering packet transmissions from Node B, as shown in Figure 1 (a), then the EEN field will be set for all the packets transmitted by Node B, informing nodes A and H the occurrence of the event. Upon the reception of a packet from B, a Kalman filter in both nodes A and H will be created, initialized, and updated due to this indirectly sensed measurement. Nodes in the neighborhood of B will then compute the current STEP $S^{j}_{k+1|k}(\mathbf{u})$ and FEDP $S^{(j,i)}_{k+1|k}$ with respect to their sensing fields. Based on these values, the nodes will decide their appropriate duty cycle levels.

Unlike the occurrence of an event, we do not expect the disappearance of an event to trigger immediate communication traffic among nearby nodes in case of indirect sensing. Therefore, the disappearance of an event can only be inferred by the absence of packets with EEN set for a period of time. Thus, duty cycle adaptation is carried out using a *soft state* approach with a timeout. Note that since each node computes its own STEP independently based on not only its own measurements but also the messages received by its neighbors, the STEP estimated at each node will be slightly different.

3) Exponential Frame Length Adjustment: Once an event is detected and a new duty cycle level is determined, the length of a frame is incremented or decremented *exponentially.* Let T_c be the current frame length corresponding to the duty cycle level d_c, T_1 the frame length of the lowest duty cycle level corresponding to d_1, and M the base of an exponentially varying frame length. Then, T_c is one of $T_n = \frac{T_1}{M^n}$, where $n \in \{1, ..., N\}$, and $M \in \mathbb{N}^*$. Note that in DSMAC and AMAC, M is always set to two, whereas in the proposed PDCA scheme it could be any number. If M is two or three, for example, then the frame length changes by doubling or tripling. This exponentially varying adaptive frame method guarantees that any pair of nodes is able to communicate by an algorithm that finds which time slots are shared by a pair of neighboring nodes, even if the nodes operate at different duty cycles. Details of this algorithm are omitted due to space constraints.

Note that the proposed PDCA scheme allows different nodes in the network to operate under various duty cycles, resulting in heterogeneous schedules in a network. As a consequence, the active periods of neighboring nodes may not overlap, meaning that the period during which a given node needs to transmit a message to its neighbor may not coincide with the period the neighbor is listening to the wireless medium. To overcome this problem, we developed a mechanism that allows nodes to calculate the moments when they are allowed to transmit messages to their neighbors based solely on the knowledge of their respective schedules, that is, without resorting to additional message exchanges. Due to space constraints, however, we do not describe this mechanism in details in this paper.

D. Fast Delivery of Event-related Packets

To reduce the latency in the delivery of information about an event to the base station, we explicitly distinguish the intermediate nodes along the routing path used to deliver this information. As we previously discussed, event-detecting nodes are identified by setting the EEN bit in the MAC header of outgoing packets. To indicate that a node is routing event-related information to the base station, on the other hand, we define the *Explicit Event-Routing Notification* (EERN) bit in the MAC header. When a node is the intended recipient of a packet in which the EEN or EERN bit is set, it implies that a prioritized flow that contains information about an event is being routed through the node. Thus, upon the reception of this packet, the node sets the EERN bit for all packets to be transmitted in the future. As long as a node is part of a routing path, it increases its duty cycle to a pre-defined level $d_{routing}$ to minimize the end-to-end latency. For example, $d_{routing}$ could be set to the maximum duty cycle d_N. Membership of a node to a routing path is also maintained as a soft state with a timeout.

IV. Performance Evaluation

In this section we demonstrate the effectiveness of the proposed PDCA scheme by evaluating the performance of the state-of-the-art MAC protocol and comparing with the performance of a new MAC protocol which includes the proposed PDCA scheme. We chose TMAC [6] as the base MAC protocol for our evaluation, since it is known to be a well-performing synchronous MAC protocol that allows active time adaptation. However, TMAC does not allow for frame length adaptation, and more importantly, no prediction is involved in the adaptation. Rather, TMAC reacts to the current network condition. By applying the PDCA scheme to TMAC, the frame length also becomes dynamic and duty cycle adjustments become predictive, resulting in better adaptivity without any design conflict. This modified TMAC will be called *P-TMAC*.

Before going into details, we would like to make a note that our proposed approach may not perform well in extreme cases in which the Kalman filter-based estimation is not reasonably accurate because of severely low sensing resolution. Such conditions can be found in extremely sparse networks. In addition to the spatial sensing resolution, the estimation accuracy would be low if the sampling frequency is too low compared to the mobility of an event.

The mobility of typical events must be taken into consideration when choosing proper PDCA parameters such as the motion model uncertainty and the STEP prediction interval. In this paper, motion uncertainty is empirically determined and the time period for STEP prediction is set to be identical to the base frame length. Optimizing these parameters according to the mobility of particular event types is part of future investigation.

We evaluate P-TMAC in the context of target tracking using the Castalia simulator [18] which is based on OMNeT++ . We simulate a network consisting of 200 TelosB nodes equipped with cameras deployed randomly under the ceiling viewing downwards in a $200m \times 200m$ area. The sensing range of each camera is a circle with a radius of $40m$. A randomly

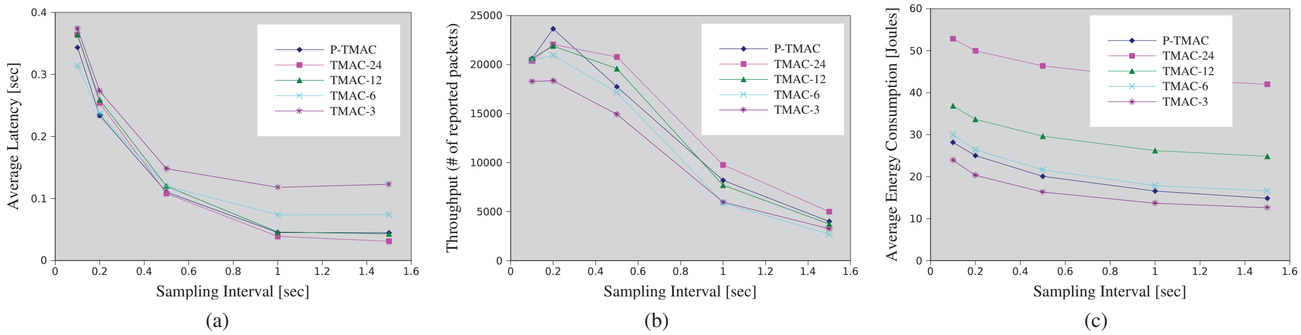


Figure 2: Simulation results of network performance in terms of (a) latency, (b) throughput, and (c) energy consumption of P-TMAC and TMAC with four different duty cycles.

Tx range	$\sim 100meters$	SYNC	$22Bytes$
Tx power	$42.24mW$	RTS/CTS	$14Bytes$
Rx power	$38mW$	ACK	$14Bytes$
Sleep power	$15\mu W$	DATA	$44Bytes$
Idle power	$3mW$	Sim. time	$2400Sec.$

Table I: Summary of simulation parameters.

moving object is assumed to exist in the network during one third of the total simulation time.

We compare the performance of P-TMAC with that of TMAC. The base frame length of P-TMAC is set to $T =$ 1000ms, its active period to 30ms, and its frame length is allowed to vary among $N = 4$ levels, corresponding to T, $T/2$, $T/4$, and $T/8$, that is, $M = 2$. Since the active period remains constant, these frame lengths correspond to duty cycles of 3%, 6%, 12%, and 24% respectively. To provide a fair comparison, we evaluate T-MAC operating at the same four duty cycles. In our experimental results, these different TMAC instances are identified as TMAC-3, TMAC-6, TMAC-12, and TMAC-24. The detailed parameters used in our evaluation are summarized in Table I.

Energy efficiency is evaluated based on the energy consumed only by the radio. To capture the performance characteristics of the MAC protocols in a collaborative processing scenario, we employ a QoS metric specifically designed for performance evaluation in event-driven WSNs, called time-bounded parameter-estimation accuracy (TIBPEA) [19]. TIBPEA is a QoS evaluation metric designed based on the fact that the greater the time-bounded reliability with which the neighboring nodes can communicate with each other, the greater the accuracy of any parameter that must be computed collaboratively. TIBPEA corresponds to the average percentage of neighbors that successfully reply to a broadcast message within a certain timeout period [19].

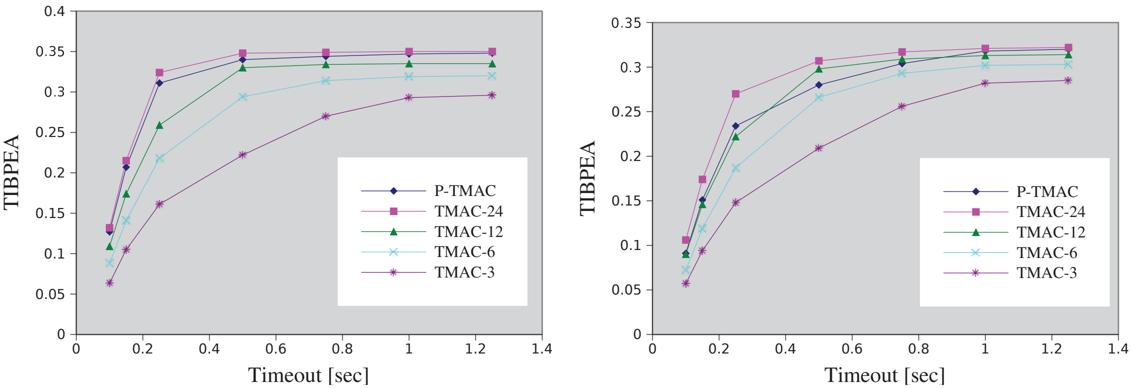
A. *Individual Processing & Reporting Scenario*

In this scenario, each node continuously senses the environment according to its own sensing interval. Since the nodes have sensing fields overlapped with others, the event may trigger multiple flows to the sink in the vicinity of the event, each flow being directed by a simple tree-based routing protocol.

Because of the adaptive frame length design, P-TMAC is expected to show performances in between TMAC-3 and TMAC-24. Fig. 2(a) shows that the latency of P-TMAC is very low, being comparable to that of TMAC-24 at different sampling intervals. Fig. 2(b) shows the throughput evaluation results. Obviously, shorter sampling intervals entail higher packet rates. To interpret this result, let us define the period from when an event enters the sensing field of a node to when it leaves it as the *sensing round*. Let us also define the first packet transmitted during each sensing round as the *link initializing packet*. With the same object motion, higher sampling rate causes more packet generation per sensing round, resulting in a low proportion of link initializing packets to the overall number of packets. TMAC is designed to work best when the rate of link initializing packets is low because of low sampling interval or slow object movement. As we can see in Fig. 2(a), the average per-hop latency of TMAC-3 increases as the sampling interval increases while P-TMAC retains its performance similar to that of TMAC-24. Providing performance comparable to TMAC-24 in terms of latency and throughput, P-TMAC still achieves energy efficiency between TMAC-3 and TMAC-6, as shown in Fig. 2 (c). It implies that P-TMAC substantially improves the tradeoff between energy and latency when compared to TMAC. For less common event types in which the event duration is less than one third of the entire simulation time, this improvement would be more evident.

B. *Collaborative Processing & Reporting Scenario*

When a node detects an event in this scenario, it first broadcasts a request message to its neighbors and collects their measurements to obtain more in-depth understanding of the event by collaborative data processing, which is usually involved in distributed algorithms. We conduct two sets of simulations with average target speeds of 6m/s and 24m/s. In each set, TIBPEA is measured with different timeout bounds. In all simulations, when the timeout bound is tight, the performance of P-TMAC is comparable to that of TMAC-24, as shown in Fig. 3. When the timeout bound is loose, P-TMAC still shows better performance than TMAC-3 but worse than TMAC-24.



(a)

(b)

Figure 3: Simulation results of TIBPEA with different average target speeds: (a) 6m/s and (b) 24m/s.

This is caused by inherent additional communication overhead of P-TMAC for broadcasting SYNC messages whenever a duty cycle adaptation occurs. Nonetheless, the superior performance of P-TMAC for delay-critical applications satisfies our design goal.

V. Conclusion

We have presented a predictive duty cycle adaptation scheme (PDCA) suitable for event-driven WSNs that *actively* adapts the duty cycle of nodes by predicting the probability that an event will occur at a node in the future. This probability is estimated from the future state of an event predicted by a Kalman filter by taking measurements from direct/indirect sensing. We proved using simulations of object tracking scenarios in realistic environments that our approach outperforms TMAC in terms of TIBPEA and presents a better tradeoff between energy efficiency and throughput or latency. We found that our approach is superior to others especially when the mobility of an event is large and when the latency requirement is strict. Predictive adaptation of other system parameters such as the camera sensing rate would be another future improvement since image capturing and processing is another major energy consuming operation in WCNs.

Acknowledgment

This work was supported by Olympus Corporation.

References

[1] I. Rhee, A. Warrier, M. Aia, J. Min, and M. L. Sichitiu, “Z-MAC: a hybrid MAC for wireless sensor networks,” *IEEE/ACM Trans. Netw.*, vol. 16, no. 3, pp. 511–524, 2008.

[2] M. Buettner, G. V. Yee, E. Anderson, and R. Han, “X-MAC: a short preamble mac protocol for duty-cycled wireless sensornetworks,”

[3] J. Polastre, J. Hill, and D. Culler, “Versatile low power media access for wireless sensor networks,” in *SenSys ’04: Proceedings of the 2nd international conference on Embedded networked sensor systems*, (New York, NY, USA), pp. 95–107, ACM, 2004.

[4] A. El-Hoiydi and J. Decotignie, “WiseMAC: An ultra low power MAC protocol for multi-hop wireless sensor networks,” *Lecture Notes in Computer Science*, vol. 3121, pp. 18–31, 2004.

[5] W. Ye, J. Heidemann, and D. Estrin, “An energy-efficient MAC protocol for wireless sensor networks,” in *IEEE INFOCOM 2002. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings*, vol. 3, 2002.

[6] 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*, pp. 171–180, ACM New York, NY, USA, 2003.

[7] Y. Nam, H. Lee, H. Jung, T. Kwon, and Y. Choi, “An adaptive MAC (A-MAC) protocol guaranteeing network lifetime for wireless sensor networks,” in *Proceedings of 12th European Wireless Conference (EW 2006), Athens, Greece*, 2006.

[8] P. Lin, C. Qiao, and X. Wang, “Medium access control with a dynamic duty cycle for sensor networks,” in *2004 IEEE Wireless Communications and Networking Conference, 2004. WCNC*, vol. 3, 2004.

[9] S. Liu, K.-W. Fan, and P. Sinha, “CMAC: An energy-efficient mac layer protocol using convergent packet forwarding for wireless sensor networks,” *ACM Trans. Sen. Netw.*, vol. 5, no. 4, pp. 1–34, 2009.

[10] K. Jamieson, H. Balakrishnan, and Y. Tay, “Sift: A MAC protocol for event-driven wireless sensor networks,” *Lecture Notes in Computer Science*, vol. 3868, p. 260, 2006.

[11] V. Namboodiri and A. Keshavarzian, “Alert: An Adaptive Low-Latency Event-Driven MAC Protocol for Wireless Sensor Networks,” in *IPSN ’08: Proceedings of the 7th international conference on Information processing in sensor networks*, (Washington, DC, USA), pp. 159–170, IEEE Computer Society, 2008.

[12] H. Medeiros, J. Park, and A. C. Kak, “A Light-Weight Event-Driven Protocol for Sensor Clustering in Wireless Camera Networks,” in *First ACM/IEEE International Conference on Distributed Smart Cameras, 2007. ICDSC ’07*, pp. 203–210, Sept. 2007.

[13] H. Medeiros, J. Park, and A. C. Kak, “Distributed Object Tracking Using a Cluster-Based Kalman Filter in Wireless Camera Networks,” *IEEE Journal of Selected Topics in Signal Processing*, vol. 2, no. 4, pp. 448–463, 2008.

[14] A. Doucet, S. Godsill, and C. Andrieu, “On sequential monte carlo sampling methods for bayesian filtering,” *Statistics and Computing*, vol. 10, pp. 197–208, 2000. 10.1023/A:1008935410038.

[15] C. Hue, J. Le Cadre, and P. Pérez, “Tracking multiple objects with particle filtering,” *IEEE Transactions on Aerospace and Electronic Systems*, vol. 38, no. 3, pp. 791–812, 2002.

[16] G. Welch and G. Bishop, “An introduction to the Kalman filter,” *University of North Carolina at Chapel Hill, Chapel Hill, NC*, 1995.

[17] B. Kusy, P. Dutta, P. Levis, M. Maroti, A. Ledeczi, and D. Culler, “Elapsed time on arrival; a simple and versatile primitive for canonical time synchronisation services,” *Int. J. Ad Hoc Ubiquitous Comput.*, vol. 1, no. 4, pp. 239–251, 2006.

[18] A. Boulis, “Castalia: revealing pitfalls in designing distributed algorithms in WSN,” in *SenSys ’07: Proceedings of the 5th international conference on Embedded networked sensor systems*, (New York, NY, USA), pp. 407–408, ACM, 2007.

[19] P. J. Shin, J. Park, and A. C. Kak, “A QoS Evaluation Testbed for MAC Protocols for Wireless Camera Networks,” in *First ACM/IEEE International Conference on Distributed Smart Cameras, 2007. ICDSC ’07*, pp. 235–242, Sept. 2007.