ActiS en: Activity-aware Sensor Network in Smart Environments[☆]

Debraj De[a,*], Shaojie Tang[b], Wen-Zhan Song[a], Diane Cook[c], Sajal Das[d]

[a] *Sensorweb Research Laboratory, Department of Computer Science, Georgia State University*
[b] *Department of Computer Science, Illinois Institute of Technology*
[c] *Artificial Intelligence Laboratory, School of Electrical Engineering and Computer Science, Washington State University*
[d] *Department of Computer Science and Engineering, University of Texas at Arlington*

Abstract

A sensor network, unlike a traditional communication network, provides high degree of visibility into environmental physical processes. Therefore its operation is driven by the activities in the environment. In long-term operations, these activities usually show certain patterns which can be learned and utilized to optimize network design. However, this has been almost unexplored in the literature. In this paper we present the design and validation of *ActiS en* system, an activity-aware sensor network in smart environments. *ActiS en* consists of three components: activity-aware sensing, activity-aware radio duty-cycling, and activity-aware and energy balanced routing. Experimental results from real testbed experiments and sensor network simulator TOSSIM validate that the activity-aware design of *ActiS en* outperforms existing methods in terms of: resource utilization (energy efficiency, lifetime etc) and system performance (data delivery throughput, delivery latency etc).

Keywords: wireless sensor networks, activity-aware, smart environment, duty-cycling, information gradient, sensing, routing, Atkinson Index, TinyOS

1. Introduction

Wireless sensor networks have enabled many important social and scientific applications and its protocol design has received considerable research interest. But many existing works did not realize an important difference between sensor networks and traditional networks. Unlike a traditional communication network, a sensor network is deeply embedded in environments and its operation is driven by the activities in the environment. In many applications such as Smart Environments, the information of event activity shows certain patterns in long run (as shown in Figure 1). Most of the sensor network designs till date under-utilized the activity pattern for performance improvement of the network. To note that in this paper activity is meant by the events that are sensed and reported by the nodes in sensor network. For example in a Smart Home environment, one type of activity is the motion activity of the residents. There remains a

[☆] This work is supported by NSF NetSE program under the research grant CNS 0914371.
[*] Corresponding author
Email addresses: dde1@student.gsu.edu (Debraj De), stang7@iit.edu (Shaojie Tang), wsong@gsu.edu (Wen-Zhan Song), djcook@wsu.edu (Diane Cook), das@cse.uta.edu (Sajal Das)

missing link in sensor network design: *the feedback from sensed and analyzed activity pattern, back to the network operation for resource usage*. The activity pattern information, if utilized in an intelligent manner, can improve the sensor network performance while reducing resource usages.

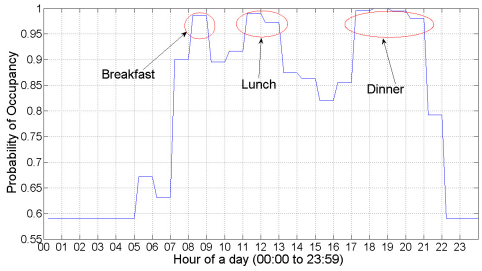


Figure 1: Probability of occupancy in the kitchen of a smart home for assisted living (CASAS [1]), detected by motion sensor

In this paper we present a novel activity-aware sensor network system called *ActiS en*. The goal is to imbue sensor networks with cognitive capabilities and activity context awareness, in order to make them act in a more intelligent manner and improve their performance. Utilizing a probabilistic activity transition graph, the *ActiS en* system uses behavioral pattern information of the sensed events. This knowledge is used efficiently to optimize performance and constrained resource usage of the sensor network. Through activity-aware sensing, activity-aware radio duty-cycling, and activity-aware and energy balanced routing, *ActiS en* can adapt its network operations based on learned activity patterns. As a case study, we have evaluated *ActiS en* based on data trace from a Smart Environment testbed.

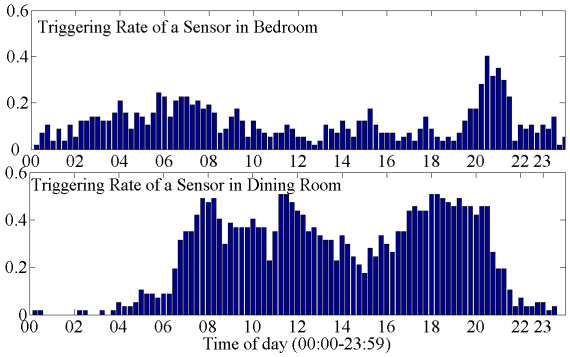


Figure 2: Change of activity pattern in 24 hours with time and space

A Smart Environment may contain many highly interactive and embedded devices, that will assist the health and safety needs of its residents or manage building energy efficiency etc. But if these sensors are continuously operating in full-alert mode, they will expend a great deal of energy and bandwidth. The result is an expensive infrastructure that requires constant maintenance to replace batteries and meet application performance. Figure 2 shows the daylong average triggering rate of two motion sensors (one in bedroom and another in dining room) in CASAS Smart Home testbed [1]. The triggering rates for two motion sensors are calculated using 24

hour data from 57 days. It is observed that there exist some activity patterns in the context of time and space. Then the sensing and communication resource usage in such sensor network can be adapted using such activity pattern. The activity-aware design of *ActiS en* tries to intelligently adapt the network resources using intelligence from activity pattern, without compromising the performance of the application it serves.

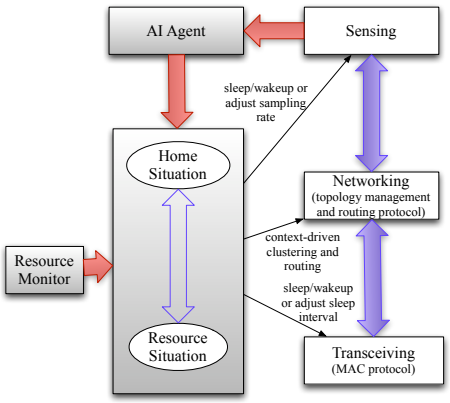


Figure 3: The system architecture of *ActiS en*

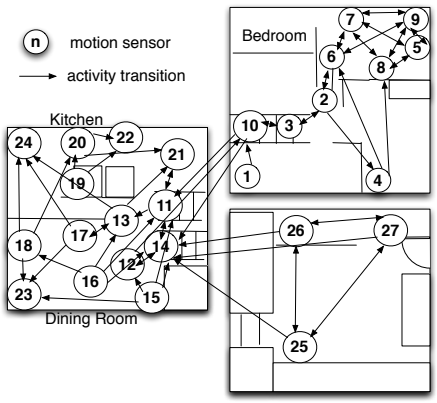


Figure 4: Activity Transition Probability Graph with 27 nodes, learnt from the CASAS Smart Home testbed [1]. Transition probability for example from node 27 to nodes 14, 25 and 26 are 12%, 45% and 40% respectively

ActiS en adapts sensing, radio duty-cycling and routing according to its historical activity pattern or transition information. Figure 3 presents the whole system architecture of *ActiS en*. An AI agent is trained from the regular activities and situations in the application environment. The agent provides *Activity Transition Probability Graph* (*AT PG*) (an example shown in Figure 4), which contains information about transition probability and transition duration of all the activities, predicted in the context of time and space. Figure 4 shows the *AT PG* with 27 nodes, learnt from the CASAS Smart Home testbed ([1]). Using such intelligence about predicted ac-

tivities, the sensing, radio duty-cycling, and routing are dynamically configured for optimized operations. The design of *ActiS en* includes:

- An activity-aware sensing scheme that achieves high event detection accuracy while reducing energy consumption through adaptive sampling intervals.
- An activity-aware radio duty-cycling protocol that dynamically adapts the radio's duty-cycle for low latency delivery, while maintaining high energy efficiency.
- An activity-aware and energy balanced routing protocol, that jointly considers activity patterns and residual network energy to balance energy consumption rate across the network thus prolonging network lifetime.

The sensor network simulation and real sensor network testbed experiments were conducted to validate the performance improvement of *ActiS en* due to the activity-aware design.

The rest of the paper is organized as follows. In section 2 we discuss the related works. In section 3 we present the protocol and system design of *ActiS en*. In section 4 we present the system evaluation and performance analysis of *ActiS en*. Finally in section 5 we conclude this work.

2. Related Work

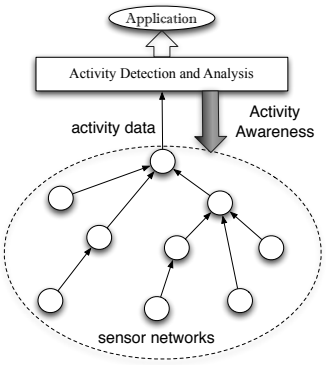


Figure 5: Activity-awareness for sensor network

There is considerable work on sensor networks deployments in Smart Environments, such as CASAS [1], MavHome [2], BScope [3], ALARM-NET [4], Gator Tech Smart House [5], Smart Medical Home [6], Spinner [7] etc. The work in CASAS [1] and MavHome [2] uses motion and other kinds of sensors for tracking and monitoring the Activities of Daily Living (ADL) for assisted living. BScope [3] presents a sensor network architecture design for activity recognition and analysis. The focus of BSCOPE is to analyze and interpret the collected sensor data using a sensory grammar. ALARM-NET [4] presents the implementation of a wireless sensor network for assisted living and residential monitoring. This work focuses on tiered network architecture with upper layer rich in energy and processing capability, and lower layer with deployed environment sensors and body worn medical sensors. This work also discusses the querying system

and security issues for smart home sensor network. Gator Tech Smart House [5] deploys Atlas sensor-actuator platform for behavioral monitoring and alteration for diabetic and obese Individuals. This also presents middleware design in general for smart spaces. The Atlas platform consists a modular architecture for scalable use of sensors and actuators. The work in [8] uses data from a deployed system to show the vulnerability of daily in-home activity information from a wireless snooping attack, called FATS attack. Besides typical sensor nodes, RFID based sensor network [9] is also being used for daily activity recognition. This work uses Intel Wisp RFID sensors emplaced on objects of daily use, for capturing daily activity patterns. Camera sensor network is used in [10] for research on vision-based reasoning for smart environments and ambient intelligence. But most of these works deal with extracting activity patterns and detecting different events for automating and facilitating application services e.g. assisted living in smart home. But in all these, detected activity patterns are not utilized back (as explained in Figure 5) in the form of activity-awareness for optimizing network operations and resource usage. Context aware networking is studied in some literature. The work in [11] presents a proactive communication algorithm for context aware sensor network. A framework for integrated unicast and multicast routing in context-aware ordered meshes is presented in [12]. But utilizing activity awareness for energy efficient and resource optimizing networking is not explored in these works, while *ActiSen* attempts these issues in depth. To best of our knowledge, activity-aware networking is almost an unexplored area in the literature. *ActiSen* system performs activity-aware networking by improving three components: sensing, radio duty-cycling and routing.

Radio duty-cycling MAC (Media Access Control) protocols have been investigated extensively for energy conservation in sensor networks. Those MAC protocols can be categorized into two types: synchronous and asynchronous approaches. Synchronous MAC protocols specify the period of wake-up and sleep for communication to reduce the unnecessary time and energy wasted in idle listening. Nodes periodically exchange SYNC packets for synchronization and data transfer in the common active schedule. S-MAC [13], T-MAC [14] etc. are examples of synchronous MAC schemes. The other type, the asynchronous MAC protocols have no control overhead for synchronization unlike synchronous schemes, in order to improve the energy efficiency compared. Examples of asynchronous schemes are B-MAC [15], X-MAC [16] etc. They rely on low power listening (LPL), also called channel sampling, to let a sender communicate to a receiver which is duty cycling. B-MAC utilizes a long preamble to achieve low power communication. In X-MAC, short preambles with target address information are used to reduce the excessive preamble, instead of a long preamble. When a receiver wakes up and detects a short preamble, it looks at the target address that is included in the preamble. If the node is the intended recipient, it keeps awake for the incoming data, otherwise it goes to sleep immediately. In this paper, we propose an activity-aware adaptive radio duty-cycling protocol *ActDutyCycling* to reduce data delivery latency while maintaining high energy-efficiency. It utilizes Information Potential or Information Gradient [17, 18, 19] to dynamically configure duty cycle of nodes in the network. Note that here we do not propose a new MAC protocol, instead, we propose an activity-aware duty-cycle adjustment algorithm that can run on top of any of those existing duty-cycling MAC protocols. *ActSee* [20] is a radio duty cycling algorithm for activity-aware sensor networks, where the routing paths of all nodes are fixed and there is a fixed network lifetime constraint. In this work we have proposed *ActDutyCycling* algorithm to adjust radio duty-cycles (influenced by activity context) of any duty-cycled MAC protocol. *ActDutyCycling* is not an independent MAC protocol itself, but it operates on top of any selected MAC protocol.

Some relevant works on energy-balanced or lifetime-maximized routing design issues include [21], [22], [23]. The work in [21] has formulated the lifetime maximization problem ass

a linear program and has solved it using distributed heuristics technique. But this work assumes that the message generation rate at nodes are fixed and known. In [22] the observation has been made that the linear program is equivalent to that of a maximum concurrent flow problem. The algorithms proposed in [21] and [22] are able to determine how the traffic (generated at a constant rate) should be split among the different routes in order to maximize the network lifetime. Since the traffic generation rates are assumed to be constant and known in these works, the network can solve the energy aware routing problem off-line. The work in [24] converts the maximum network lifetime problem into a utility-based nonlinear optimization problem and proposes a distributed routing algorithm to solve the problem. This work also assumes that the data generation rate at each node is fixed and known in advance. In event-driven sensor network applications (including activity detection in smart environment), the message generation rate at different nodes are non-uniform and also dynamic. Therefore there is no complete advance knowledge of future message generation process. This problem needs to be solved with online routing protocol which does not need to know the message generation rates. Our proposed *ActRouting* is an online routing protocol. An online routing algorithm max-min zP_{min} is proposed in [23] for network lifetime maximization and it provides a competitive analysis. $CMAX$ [25] proposes an algorithm that tries to maximize the network capacity using shortest path computation with routing metric based on node remaining energy. The work in [26] proposes $E - WME$ online routing algorithm for the scenario of energy harvesting sensor nodes. Most of these energy aware online routing algorithms are based on remaining energy of relaying nodes. These works don't try to maintain energy balance in the network as a whole, and don't learn from activity patterns. An energy-welfare index (using Atkinson Inequality Index) is utilized in [27] to keep energy balance in network. But the forwarder selection procedure is complex and expensive (computing the index for each forwarder for each packet). But energy balance index in *ActRouting* is localized and simple to compute. TITAN [28] presents a proactive topology management algorithm. But it does not provide any explicit energy balancing, rather achieves implicit energy efficiency from delayed RREQ (route request message). Our *ActRouting* proposes an explicit energy balanced data delivery protocol based on degree of energy balance in local neighborhood in the network. Also to note that, for the goal of maximizing network lifetime one possible solution may be to route the messages along the path with the maximal minimal fraction of remaining energy (the max-min routing). The performance of max-min path can degrade in situations (as described with some example in [23]). Another issue with the max-min routing is that following route with max-min energy node may be expensive compared to other possible paths. For large number of data streams there can be significant energy consumption for common nodes on max-min routing paths. This creates bottleneck nodes with very high energy consumption and thus degrades the network lifetime quickly.

3. *ActiSen*: Activity-Aware Sensor Network System

In a Smart Home at certain time of the day, a resident's Activities of Daily Living (ADL) (say activity A_i) in a region is monitored by a set or cluster of sensors, say C_i. It is worth noting that *ActiSen* system and its protocols are designed for not only single source of activity (e.g. single resident), but generally for multiple activities (e.g. multiple smart home residents). Then in the smart home scenario, after some time the resident can probabilistically move to either cluster C_j or C_k or some other cluster. The activity-aware sensor network in such scenario can intelligently utilize its resource using the knowledge from an Activity Transition Probability Graph or $ATPG$ (say G_{act}). Each node of G_{act} denotes some specific cluster C_i. The edge from node C_i to node

C_j denotes the transition tuple $< p_{ij}, t_{ij} >$ from C_i to C_j. p_{ij} is the predicted transition probability and t_{ij} is the predicted transition time. In *ActiSen*, the graph G_{act} is learned and updated by an AI agent (which is beyond the scope of this paper).

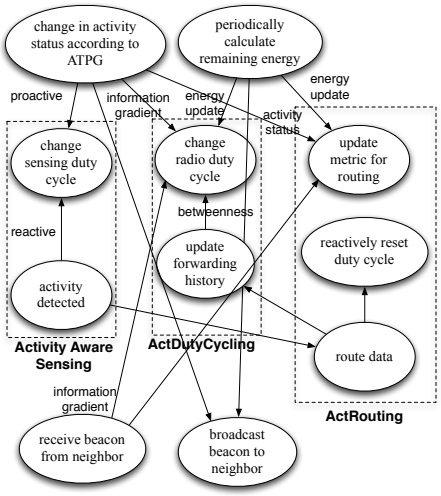


Figure 6: Workflow diagram of *ActiSen* system and inter-relationship among (a) Activity-Aware Sensing, (b) *ActDutyCycling*, and (c) *ActRouting*

After learning the ATPG G_{act}, the activity-aware sensor network utilizes it to optimize its sensing, radio duty-cycling and routing operations. The protocols in *ActiSen* are designed in such a way that activity detection sensors and the wireless radio can sleep as much as possible, while the activity events are reliably covered, detected and reported to the sink (i.e. the base station). Now we present an overall description of how the network in *ActiSen* works for the example of smart home scenario. All the activity-aware protocols in *ActiSen* are later described in detail.

The design of sensing, radio duty-cycling, routing and their inter-relationship in *ActiSen* system is shown in Figure 6. Each of the algorithms are later described in details in next subsections. Here is a brief description of computation overhead of different components (sensor nodes and sink). (i) The event detection data in network is collected in sink node for application purpose, and used for constructing ATPG activity pattern. The activity pattern analysis is only done in sink. The activity transition pattern is disseminated into the network only once and updated if any change happens in pattern, which is very rare. (ii) All other calculations involved in *ActiSen* are distributed and localized in the network. The sensor duty cycle calculation uses locally stored activity information. *ActDutyCycling* uses information of own and neighbors to calculate local node radio duty cycle. *ActRouting* also uses own and neighbors information only for deciding next hop node relay node. So all computations, except activity pattern analysis are distributed and localized in the network. This makes it practically applicable to networks independent of size.

Here is a brief description of communication overhead of different components (sensor nodes and sink). (i) All the sensor nodes in the network continuously sense activity and send the activity data through multi-hop network data collection to the sink node. This is the convergecast or

data collection communication that goes on continuously whenever activity event is triggered. (ii) The sensor nodes, in their local 1-hop neighborhood, periodically exchange (through 1-hop broadcast communication) beacon message and share node or local neighbor information. (iii) Each node also periodically sends (network-wide data collection communication) one local status data packet (containing information of remaining energy, duty cycle, hop count etc.) to sink node with a relatively large period. (iv) If there is any major change in the analyzed activity transition pattern in the sink, the new activity transition information is disseminated into the network from the sink node (through data dissemination communication). In *ActiSen* system it is not needed frequently to disseminate data into network from sink. The dissemination can use any standard data disseminating protocol (e.g. *Cascades* [32]).

From the activity transition probability graph G_{act} for current activity, a cluster C_i of sensor nodes is constructed. The selection of member nodes of C_i is determined by the location context of activity. Then the nodes which are not member of C_i can turn their activity detection sensors ON less frequently. The nodes in C_i turn their activity detection sensors ON more frequently. Example of cluster is kitchen cluster where resident activities can occur for considerable time at certain phases of ADL (Activities of Daily Living). Now according to the activity transition graph, the active cluster is changed from C_i to next active cluster C_j, for the next predicted activity context. After the predicted transition time t_{ij}, C_i triggers C_j to wake up and reconfigure. The previous cluster C_i can be kept ON for a marginal time to watch for any remaining activity. Now we explain in details, how sensing, radio duty cycling and routing is performed in *ActiSen* system.

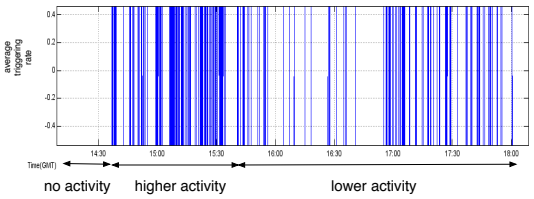


Figure 7: Changing rate of activity detected; PIR motion sensor data (sampling frequency 10 Hz) shown with transition from no activity to higher activity and then to lower activity

3.1. Activity-Aware Sensing

Figure 7 shows an example of detected PIR motion sensor signal for a typical transition of human activity among different intensities. Maintaining a fixed sampling interval for sensing could result in either high energy consumption (when interval is small) or low detection accuracy (when interval is big). Motivated by the need of an activity-aware adaptive sensing scheme, we have proposed the following algorithm. Each node in the network proactively and reactively adapts the sampling intervals.

Proactive selection of base sampling interval: As described in Section 3, each sensor node belongs to one of the three sets regarding role in activity sensing: *Sleep* (nodes outside active or next predicted clusters), *Quasi-Active* (nodes inside the next predicted clusters and outside active clusters), *Active* (nodes inside the active clusters). Then the sampling interval for activity sensing is dynamically configured based on the type or role of the node at current moment.

Algorithm 1 *ActSensing*: Activity-aware Adaptive Sensing

Each node i dynamically adjusts sampling intervals as follows:

loop
  if $i \in (Sleep|Quasi-Active|Active)$ **then**
    $T_{OFF} = (T^s_{OFF}|T^q_{OFF}|T^a_{OFF})$
  end if
  while *no source transition* **do**
    if $T^{peak}_{min}(t) < T^{peak}_{min}(t-1)$ **then**
      $DC(t+1) = (DC(t) + \delta)$
    else
      if $T^{peak}_{min}(t) > T^{peak}_{min}(t-1)$ **then**
        $DC(t+1) = (DC(t) - \delta)$
      else
        $DC(t+1) = DC(t)$
      end if
    end if
  end while
end loop

The activity detection sensor is ON for T_{ON} and is then OFF for time T^s_{OFF} (for $Sleep$) or T^q_{OFF} (for $Quasi-Active$) or T^a_{OFF} (for $Active$), such that $T^s_{OFF} > T^q_{OFF} > T^a_{OFF}$. This technique lets only the nodes in active regions sense more frequently, and the other nodes sense less frequently, thus saving energy. In addition to wake and sleep schedule for sensor, the nodes keep two more timing information. Once the sensor is ON, it only samples it after a stabilization delay T_{STAB} to remove initial erroneous samples. Also the sensor is kept ON for an elastic margin time for allowing certain fixed N_{SAMPLE} number of samples in case a motion is detected inside T_{ON}. N_{SAMPLE} samples are needed by application for detecting level of activities. Thus adaptive sampling interval is needed for the purpose of capturing important motion activity.

Reactive adjustment of sampling interval: In addition to proactively configured base sampling interval for sensing, the proposed design lets the individual nodes reactively adjust the sensing frequency in a finer scale with intensity of activity. When the activity frequency is low, a higher sampling interval is fine for detecting the events. But when activity frequency goes high, the sensing interval can be adaptively decreased to enable successful event detection. This allows the nodes to potentially save more energy in idle situation, but also remain adaptive to changing activity level. Let $T^{peak}_{min}(t)$ is the minimum timegap between two consecutive peaks in detected activity signal during t-th interval of sensing. Then from tracking $T^{peak}_{min}(t)$ during intervals, the sensing frequency $DC(t+1)$ is further adjusted with a finer scale of say δ. For example, the scale δ is 1%. The reactive adjustment of sampling interval is also useful near the transition of data source. Overall the proposed activity-aware sensing algorithm proactively and reactively adapts the sensing frequency for saving energy, and assuring reliable event detection. The *ActSensing* is formally described in Algorithm 1.

Activity event beyond prediction: The activity-aware sensing algorithm is protected from missed detection of unusual activity that don't follow the predictions in $ATPG$. T^s_{OFF} can be set to a low but safe value based on human motion frequency. Then on occurrence of such unusual event, activity-aware sensing algorithm will be able to detect it. Once detected, the

reactive adjustment property will increase the sampling frequency for finer activity detection. Alternatively *sensor wake-on* hardware property (in [29]) can be used by low energy cost sensor (always on) to wake up higher energy cost sensor to detect activities. But this alternative can be applied in sensing only some physical phenomena.

3.2. Activity-Aware Radio Duty-Cycling

In this section we describe our proposed activity-aware radio duty-cycling protocol. In a typical wireless networks, a one-hop packet delivery latency usually includes processing delay, transmission delay, and propagation delay, which are usually in milliseconds order. However, in a low-duty cycle network, a sender may need to wait for its receiver to wake up before it can send a packet. Thus sleep interval dominates the overall delivery latency. Therefore, in this paper we only consider sleep latency as notion of delay.

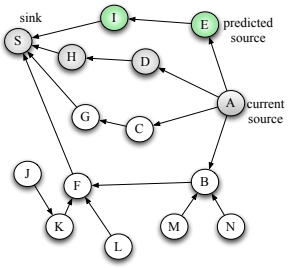


Figure 8: Scenario showing need of activity-aware radio duty-cycling

Need of non-uniform radio duty-cycling Scheduling the operation of the wireless radio is a crucial task in achieving efficient performance for wireless sensor networks. Now in the network at any time only certain active region of sensor nodes generate data. This scenario is shown by an example in Figure 8. Suppose in the network topology, the currently active data source is node A and predicted active data source is node E. To note that in general case there can be multiple current or predicted active (data generating) nodes in network. Let the probable active routing path is A-D-H-S, and the probable routing path for predicted source is E-I-S. Then an efficient radio duty-cycling policy is: (i) to maintain high duty cycle for nodes on and near the routing paths for active and predicted sources (for fast and reliable data delivery on any meaningful route selected), and (ii) to maintain low duty cycle (for energy saving) for potentially idle nodes away from active routing paths (for example nodes J, K, L, M, N etc.). The major challenge for maintaining such non-uniform duty cycling is that, the distribution of duty cycle has to dynamically adapt with change of active and predicted sources. In this aspect we have designed *ActDutyCycling*, an activity-aware radio duty-cycling protocol. It is not just more energy efficient but also provides reduced latency for delivery of data to sink. The potential of *ActDutyCycling* is that it can dynamically adapt the radio duty-cycling operation across the network, in the context of current and predicted active regions.

Before describing *ActDutyCycling* in detail, we define the parameters pertaining to the radio duty-cycling decision. The parameters for each node (say i) are described as follows:

Information Gradient (g_i): Information potential or information gradient g_i (as described in [17] and [18]) is a real valued function for each node i. It is defined as a function that meets the following requirements: its value is (a) 0 at the sink, (b) 1 at nodes on the active/predicted cluster, (c) at any other node, the function value equals the average value of its neighbors. It can be shown that there is a unique such function for any given source (it is the harmonic function meeting the specified boundary conditions). The information gradient is a 'smooth' function with no local extrema. It is stable to small changes in network connectivity. Effectively the information gradient g_i indicates the estimate of the nodes' relative position between active/predicted cluster (source of data) and the base station (sink of data). This parameter is used in *ActDutyCycling* for controlling the radio duty cycle in context of activity and node's possible role as relay. *ActDutyCycling* actually uses two gradient values: (a) information gradient for current data source (gc_i), and (b) information gradient for predicted data source (gp_i).

Hopcount (c_i): It is a representative of the depth or the minimum hopcount from node i to the base station. For scaling to maximum value of 1, c_i is the ratio of hopcount to base station (say hop_i) to the estimated depth of the total network tree (say D). Then, $c_i = hop_i/D$. For practical issues, if the depth of the network is hard to maintain, then one of the two alternative definitions can be used. (i) c_i can be considered as the ratio of hopcount to base station to the total number of nodes in network (say N). Then, $c_i = hop_i/N$. (ii) If ideally c_i has to be a local property of node, then it can be defined as c_i = (1-1/hop_i). In experiments we have used the first definition of c_i.

Betweenness(b_i): Based on the activity transition probability graph and activity-aware routing path designed for each predicted source, it is easy to calculate the probability that a given sensor i will be involved in the routing path after the next source transition. The intuition behind our design is that the sensors that have higher probability of becoming relay nodes in the next source transition, deserve higher duty-cycle. We introduce a concept, betweenness, that have been widely used in graph theory to describe the importance of a given vertex. Basically, vertices that occur on many shortest paths between other vertices have higher betweenness than those that do not. In this paper, we redefine the betweenness of a vertex as the probability that it will appear in the activity-aware routing path on the next source transition. Assume m is current source, $p_{m,j}$ is the transition probability from m to j. Let $I_j(i)$ indicate whether i will appear at the routing path from j to base station, e.g. $I_j(i) = 1$ if i will appear in the routing path and $I_j(i) = 0$ otherwise. $I_j(i)$ can be obtained immediately after the activity-aware routing path is found (as discussed in next section). Thus, the betweenness of given sensor i can be formally defined as: $b_i = \sum_{j \in V} p_{m,j} \cdot I_j(i)$.

Relative remaining Energy (e_i): It is an indicator of remaining energy of node i with respect to the nodes in its neighborhood. For scaling to maximum value of 1, e_i is represented as the ratio of remaining energy of node i to the maximum of the energy of the neighbors and itself.

Now we describe our proposed activity-aware radio duty-cycling protocol, we call *ActDutyCycling*. According to the data sending rate of the application, the radio on each node has a maximum sleep interval (say TS^{max}) and a minimum sleep interval (say TS^{min}). The naive radio duty-cycling will select a static and uniform sleep interval TS^{naive} for each node in the network ($TS^{min} \leq TS^{naive} \leq TS^{max}$). The radio ON time will be controlled by MAC protocol selected.

But *ActDutyCycling* prefers non-uniform duty cycle, and assigns duty cycle to nodes based on their expected role in data delivery and available energy. In *ActDutyCycling*, a fraction of TS^{max} is dynamically assigned as the sleep interval TS^{acti} (still satisfying $TS^{min} \leq TS^{acti} \leq TS^{max}$). This selection of TS^{acti} is updated with time, is non-uniform across nodes, and is a

function of the parameters (described above): information gradient for current source, information gradient for predicted source, hopcount, betweenness and relative remaining energy.

Determination of Sleep Interval: Suppose cluster C_i is active at current moment and cluster C_j is predicted to be the next active cluster. Then *ActDutyCycling* attempts to control the radio duty cycle of the sensor nodes in the network, in order to satisfy following requirements:

(a) Reliable and low latency delivery of sensed data from active cluster node to the base station.

(b) Assurance of future activity detection accuracy, by setting up active path in advance, from predicted cluster to the base station.

(c) Low duty cycle for nodes in inactive region for energy saving.

The effective sleep interval $TS_i^{acti}(t)$ for node i is a function f($gc_i(t)$, $gp_i(t)$, $c_i(t)$, $b_i(t)$, $e_i(t)$) such that:

(a) As information gradient gc_i or gp_i increases, TS_i^{acti} decreases. Information gradient is highest (fixed at 1) at nodes in the active and predicted clusters, and is lowest (fixed at 0) at the base station. So the radio of the nodes near active/predicted cluster are made to be active more for data relay.

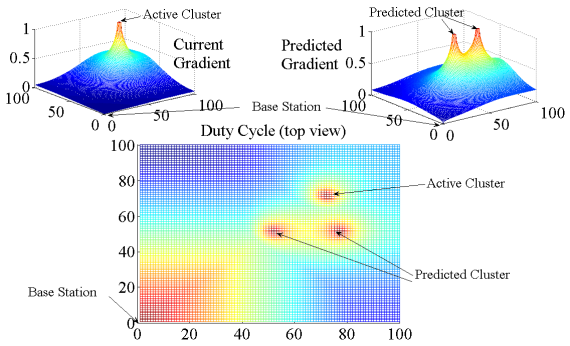


Figure 9: Activity-Aware non-uniform radio duty-cycling in network

(b) As hopcount c_i to the base station increases, TS_i^{acti} increases. So radio of the nodes near base station are made to be active more for data relay.

(c) As betweenness b_i of a node increases TS_i^{acti} decreases. So probable nodes on the routing path from next predicted source are made to be active more.

(d) If relative remaining energy e_i of node i is below a critically low threshold (say e_{low}), TS_i^{acti} increases. So radio of any node with critically low remaining energy in neighborhood can sleep more for saving energy, and try to participate less in any data delivery.

Now *ActDutyCycling* works as follows. Each sensor node i in the entire network keeps track of: hop count c_i, relative remaining energy e_i, betweenness b_i and two gradient values (gc_i for currently active cluster, gp_i for predicted cluster). The c_i, e_i, b_i, gc_i and gp_i are periodically updated in a distributed manner among the neighbors. Then each node i computes the effective sleep interval $TS_i^{acti}(t)$ as shown in Algorithm 2.

The working of *ActDutyCycling* is illustrated in Figure 10 in a network topology same as in

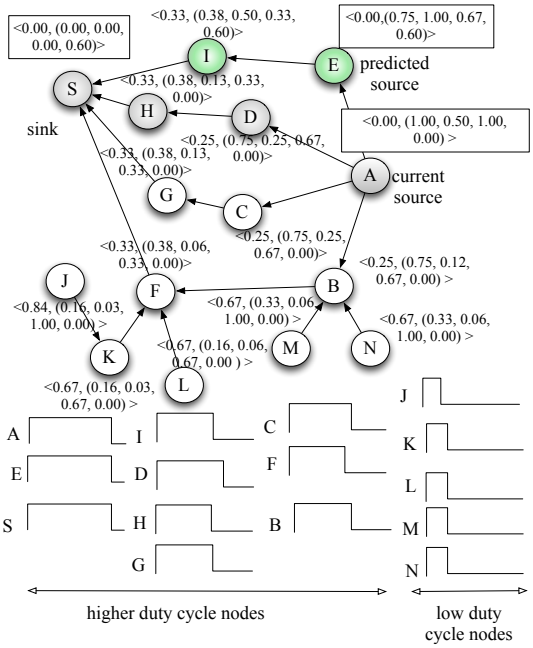


Figure 10: Non-uniform Duty cycle control in *ActDutyCycling*. The set of parameters $<f, (gc, gp, c, b)>$ for each nodes are shown. It is assumed that the predicted source transition is from node A to node E with a probability 0.60

Algorithm 2 *ActDutyCycling*: Activity-aware Radio Duty-cycling

if $i \in (sink \cup active\ cluster \cup predicted\ cluster)$ **then**
  $TS_i^{acti}(t) = TS^{min}$
else
  if $e_i(t) \leq e_{low}$ **then**
    $TS_i^{acti}(t) = TS^{max}$
  else
    f=$min((1\text{-}gc_i(t)), (1\text{-}gp_i(t)), c_i(t), (1\text{-}b_i(t)))$
    $TS_i^{acti}(t) = max(TS^{max}.f, TS^{min})$
  end if
end if

Figure 8. At certain moment in the network, the parameters of *ActDutyCycling* for each node are shown. Then using gradients, hopcount, betweenness and energy, the nodes calculate their individual sleep interval. As shown in the figure: (i) sink (S), current source (A) and predicted source (E) select minimum sleep interval (i.e. maximum duty cycle), (ii) the nodes on or near active route between current/predicted source and sink (nodes I, H, D, G, C, B and F) select relatively low sleep interval (i.e. higher duty cycle), (iii) nodes away from active region (nodes J, K, L, M and N) select larger sleep interval (i.e. low duty cycle). Here node I has a high betweenness, so selects relatively low sleep interval (i.e. higher duty cycle). Therefore in this activity context *ActDutyCycling* lets only meaningful nodes keep awake more, while letting other nodes sleep more.

The effectiveness of *ActDutyCycling* is also shown in a simulation environment of 100x100 sensor grid in Figure 9. As shown in Figure 9, the nodes in the grid maintain information gradients for currently active and predicted clusters. Then according to *ActDutyCycling* mechanism the nodes calculate their individual duty cycle. It can be clearly observed that only a set of nodes between base station and possible data source (active and predicted clusters) maintain high duty cycle, while the other nodes in network keep a relatively low duty cycle. This shows the advantage of using activity-aware radio duty-cycling, leading to energy efficiency and at the same time assuring reliable and low latency data delivery.

The role of *ActDutyCycling* in the whole *ActiSen* system and the relationship of *ActDutyCycling* with sensing and routing components is illustrated in Figure 6.

3.3. Activity-Aware and Energy-balanced Routing

In *ActiSen*, at any moment, only some set of nodes perform activity detection. This non-uniform distribution of data source in network leads to non-uniform energy consumption of nodes in the network. In such scenario an efficient routing solution is the one with: (i) awareness of activity context and (ii) maintenance of energy balance across network. The nodes in active region can be excluded from task of data relay as much as possible. This will keep the processing unit (MCU) of the active nodes available for sensing, processing and communicating. Then there are other nodes in the network available who are not performing any activity detection. These nodes have their energy and MCU available for relaying the data stream to the sink. Therefore meeting network-wide energy balance and activity-awareness are also interlinked to some extent.

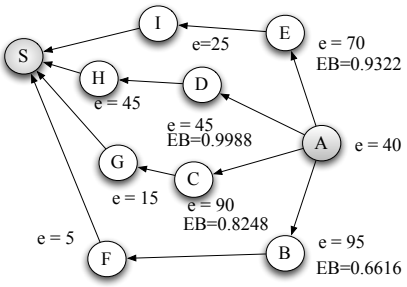


Figure 11: Energy balanced routing

Network energy balance is a critical issue in routing for achieving longer network lifetime and reduced network energy consumption. A greedy routing scheme will exploit nodes with good

path quality or node with high energy, leading to energy drain of certain nodes and eventually network failure. For example, in the network in Figure 11, nodes A, B, C, D, E, F, G, H and I have their remaining energy e as shown. Node A, the currently active data source, has to decide the relay node from B, C, D, E. An energy greedy algorithm, looking for maximum energy neighbor, will choose path A-B-F-S. Another energy greedy routing scheme, looking for path with maximum energy, will choose path A-C-G-S. But in all of these cases the node with low energy (F, G or I) will soon drain its energy, leading to network failure. But an energy balanced routing, with localized knowledge about degree of energy balance in network, is more efficient. Such energy balancing routing algorithm will choose a better energy balanced path A-D-H-S. This can increase the network lifetime considerably. An energy balanced routing is different from max-min routing. A max-min routing protocol selects a path with maximum of the minimum energy. But our energy balanced protocol prefers the local energy balance in the network for routing decision. This can be more useful also in case there is data flow from multiple sources. In that case for max-min, all the flows will be directed towards the max-min energy node. This will be detrimental to the cause resulting in increased routing stretch and increased energy consumption of some bottleneck nodes. Then instead assigning locally energy balanced path for each flow will be more effective.

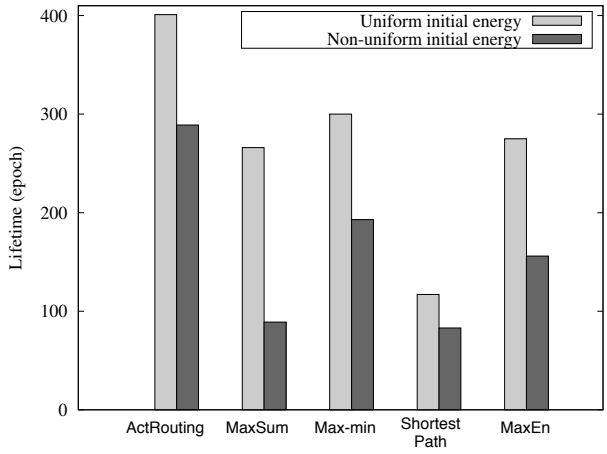


Figure 12: Network Lifetime of 30×30 grid network for different routing algorithms (MaxSum: maximum sum of energy of routing path, Max-Min: maximum of minimum energy on routing path, ShortestPath: shortest path (minimum hop) routing, MaxEn: maximum energy neighbor).

Through MATLAB simulation in large scale network, it has been confirmed that our proposed *ActRouting* performs much better than other energy aware routing techniques (maximum energy neighbor, maximum energy path, max-min energy). The simulation environment is set up as follows: in each time epoch each node in the network generates and sends a flow of packets and forwards received packets from children in last epoch. The base station is placed at one corner of the grid. The average lifetime of a number of experiment runs is shown in Figure 12. One more useful observation from large scale network simulation is that, *ActRouting* performs even better in the scenario where nodes boot up with uneven distribution of remaining energy. Therefore the proposed routing algorithm can efficiently extend the lifetime of a network even

with considerable energy imbalance.

The proposed *ActRouting* is based on activity context and energy inequality. It employs intelligently energy balanced data delivery in local region of the network, leading to global energy balance and improved network lifetime. The rationale behind using routing based on local energy balance is that, it can guide the data through energy balanced path without requiring any global knowledge or large network state. *ActRouting* also handles the situation where local region is energy balanced, but all the nodes have low energy. Atkinson's Inequality Index [30] is used in *ActRouting* for indexing local energy balance, because it uses local entropy and the index can also be controlled based on needed degree of energy balance.

Atkinson Index: In order to reach a balance in energy consumption rate across the network we use Atkinson's Inequality Index [30]. It is a measure of economic income inequality in a society. The index can be turned into a normative measure by imposing a coefficient ε to weight incomes. Greater weight can be placed on changes in a given portion of the income distribution by choosing ε, the level of *Inequality Aversion*. The Atkinson index becomes more sensitive to changes at the lower end of the income distribution as ε approaches 1. Conversely, as the level of inequality aversion falls (ε approaching 0) the Atkinson Index becomes more sensitive to changes in the upper end of the income distribution. Atkinson index AT is defined as in equation 1.

$$AT = 1 - \frac{1}{\mu}\left(\frac{1}{N}\sum_{i=1}^{N} y_i^{(1-\varepsilon)}\right)^{1/(1-\varepsilon)} \tag{1}$$

Where $0 \leq \varepsilon < 1$, y_i is the individual income of i-th entity (i = 1, 2, ..., N) and μ is the mean income of total N entities. We have used the Atkinson index AT for the scenario of sensor networks where the income parameter is replaced by the parameter normalized remaining energy $re_i(t)$, the remaining energy of node i at time t divided by the maximum energy capacity. Therefore Atkinson index AT of the entire network at time t is as shown in equation 2.

$$AT(t) = 1 - \frac{1}{re_{avg}(t)}\left(\frac{1}{N}\sum_{i=1}^{N} re_i(t)^{(1-\varepsilon)}\right)^{1/(1-\varepsilon)} \tag{2}$$

Where $re_{avg}(t)$ is the average of the remaining energy $re_i(t)$ of all the nodes in the network at time t, N is the number of nodes in network. Since this index contains information of all the nodes in the network, only a centralized algorithm can compute the index completely. But it is possible to apply a distributed protocol where every node can compute the index locally within its 1-hop neighborhood. We apply the formulated Atkinson index AT in the problem of data forwarding. When a node has data packet to forward, the research challenge is to choose the relay node from the next available neighbors. The use of Atkinson index AT penalizes large inequality in remaining energy. An energy balanced relay node selection tries to maintain a balance in energy consumption rate of the nodes in the local region. This local balance in turn results in balance in energy consumption across the whole network.

Energy Balance Metric: Now we formulate our proposed routing metric, called Energy Balance (say EB) for relay selection. We design EB as (1-AT_{local}), where AT_{local} is the Atkinson Index computed locally in 1-hop neighborhood. Since AT_{local} denotes the level of energy inequality in 1-hop neighborhood, EB here denotes the level of energy equality or energy balance

in 1-hop neighborhood. So at time t the routing metric $EB_i(t)$ computed by each node i is as shown in equation 3.

$$EB_i(t) = \frac{1}{re_{avg}(t)}\left(\frac{1}{|\Delta|}\sum_{i=1}^{\Delta} re_i(t)^{(1-\varepsilon)}\right)^{1/(1-\varepsilon)} \tag{3}$$

To note that for each node, Δ is the set of 1-hop neighbors and the node itself. So the metric EB is calculated using remaining energy information of the neighbors and the node itself.

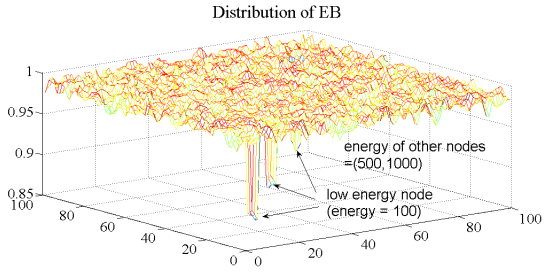


Figure 13: Distribution of Energy Balance (EB)

In Figure 13 the effectiveness of EB is shown. In a simulated environment in MATLAB with 100x100 sensor network grid, each node has maximum 8 neighbors. Only three nodes in the network have energy value of 100. But other nodes have energy between 500 to 1000. Then the distribution of locally computed EB across the network is shown in Figure 13. It can be observed that EB is high enough (very close to 1) everywhere, except in the neighboring region of the nodes with low energy. Therefore in a distributed network EB is a meaningful indicator of region with significant energy imbalance. Lower EB indicates higher degree of energy imbalance. The advantage of *ActRouting* in the initial example is described in Figure 11 with calculated EB metric.

Relay Selection: Now we describe the data forwarding scheme. The nodes in 1-hop neighborhood periodically share their normalized remaining energy (re_k) and Energy Balance metric EB_k ($k \in$ neighbor) through beacon message broadcast. From the $re_k(t)$'s of neighbors and its own remaining energy $re_i(t)$, each node i recomputes its metric $EB_i(t)$ and shares its current value of $EB_i(t)$ with neighbors using broadcasted beacon message. Suppose at time t, node i has data packet $P_i(t)$ to send to the next relay node. Then energy balanced data delivery is performed as follows. (i) Having information of EB for all the neighbors, node i chooses the neighbor node with maximum EB, which has the same or less hop count to base station, as the next relay node to forward the data to. (ii) An opposite approach is used in relay selection when the remaining energy of the node is critically low. Then node i chooses the neighbor node as relay, which has minimum EB (in a hope to find a local region with some high remaining energy nodes) and same or less hop count to base station. (iii) In case the EB of the neighbors are very close to each other, the relay is selected as the one with maximum remaining energy, which has the same or less hop count to base station.

In addition *ActRouting* uses more activity-awareness in following way. A node outside currently active or predicted active region, has higher priority of getting selected as relay node (than a node inside region of active or predicted data source). Overall, in a completely distributed

manner the proposed algorithm ensures relatively uniform distribution of energy consumption rate and hence better network lifetime. The strength of the proposed algorithm is that locally computed EB has inherent visibility into energy distribution in 2-hop neighborhood. To ensure faster routing convergence an added technique can be used. If a data packet is forwarded to nodes with same hop count for certain number of times, in the next step it is forwarded to the node with only lesser hopcount and better EB.

Algorithm 3 *ActRouting*: Activity-aware and Energy-balanced Routing

Each node i periodically computes the Energy Balance (EB) involving all of its 1-hop neighbors and itself. If node i has data packet to forward, select node j as its relay node as follows:

if $(EB_k^{min}/EB_k^{max}) \geq 0.9$ $(k \in neighbor_i)$ **then**
  $j = max(remaining\ energy)\ AND\ hop_j \leq hop_i$
else
  if $energy_i \leq energy_{critical}$ **then**
    $j = min(EB)\ AND\ hop_j \leq hop_i$
  else
    $j = max(EB)\ AND\ hop_j \leq hop_i$
  end if
end if

Readjustment of radio duty cycle: For a data forwarding node, once the neighbor relay node is selected using *ActRouting*, it will readjust the duty cycle of the relay. This can keep the selected route active so that the stream of data from the current source can be sent to base station with reliability and low latency. Then, once a node i selects its relay as node j, i repeatedly unicasts a message to j (until acknowledged) notifying it to run the radio on maximum duty cycle set by the application. To note that when current data source changes, j will not receive any data to forward for a number of radio sleep/wake intervals. Then j will run the radio back to the duty cycle decided by *ActDutyCycling*.

The role of *ActRouting* in the whole *ActiSen* system and the relationship of *ActRouting* with sensing and radio duty-cycling components is illustrated in Figure 6.

3.4. ActiSen System Design

The *ActiSen* system is designed in TinyOS-2.x. The TinyOS software architecture in *ActiSen* is shown in Figure 14. The knowledge of Activity-Awareness resides in the middleware that can be utilized by application for smart environment. The cluster membership information is configured from Activity-Awareness module to the *ActRouting* module in network layer to dynamically configure the current gradient, predicted gradient and betweenness parameters. Based on these parameters the *ActDutyCycling* module configures link layer for dynamically configuring the sleep interval. The *ActSensing* module in sensing component uses activity knowledge from Activity-Awareness module and accordingly configures duty cycle for the activity detection sensors. The energy consumption is calculated using the relevant model of radio transmission, reception and radio idle states. The values used in the energy model are as follows: CC2420 radio parameters (19.7 mA current consumption in receive mode, 17.4 mA current consumption in transmit mode), 250 kbps data rate with 48 kByte data packet size, and the MSP430 MCU parameters (3 mA current consumption in active mode due to sensing and computation). The

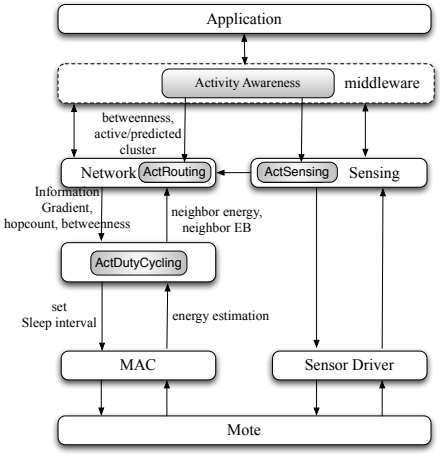


Figure 14: TinyOS software structure of *ActiSen*

remaining node energy is accordingly updated. The node remaining energy information is used in the *ActRouting* network module in terms of normalized remaining energy and Energy Balance parameters. The information gradients, hopcount, remaining energy, Energy Balance (EB) are shared among neighbors, piggybacked in the broadcasted beacon message. Instead of beacon message of MAC protocols, the software design uses beacon message in routing protocol layer (in route and neighbor maintenance routine). The beacon message is broadcasted with (adaptive time period) among the neighbors. We have utilized the existing beacon message used in the routing layer. These beacon messages are piggybacked with more information containing information gradient, hop count and other required parameters. The beacon messaging in *ActiSen* uses dynamic and adaptive beacon period.

4. Performance Evaluation and Analysis

In this section we represent in detail the experimental evaluation and performance analysis to show the effectiveness of our proposed activity-aware design of *ActiSen*.

4.1. Experiment in Large Scale Real Sensor Network Tested

Evaluation environment: We have evaluated our proposed *ActiSen* system in large scale 82 node network of $TelosB$ motes in Harvard *Motelab* sensor network testbed [31]. The experiments are conducted in 82 node network physically distributed in three floors, as shown in Figures 15(a), 15(b) and 15(c). The default radio range of used $TeloB$ motes (uses CC2420 radio chip) is typically 50m for indoors and 125m for outdoors. The experiment environment in *Motelab* testbed is indoor environment.

Activity transition and data generation: From a separately deployed motion sensor network testbed we have learned the activity transition patterns and have validated the construction of activity transition graph $ATPG$ (Figure 4). The activity transition patterns are modified to be

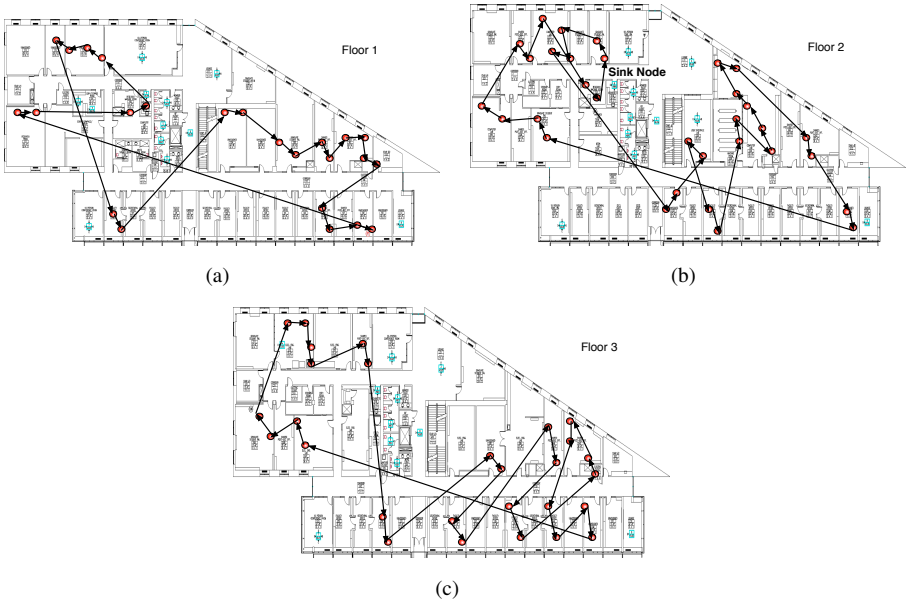


Figure 15: Most frequent activity sequences (order of active nodes) occurred in each floor of Motelab testbed during experiment

scalable for a 82 node Motelab testbed, and is injected in the testbed for activity event generation and activity transition. The activity transition decides the order with which nodes will be active.

The activity event generation makes node(s) active, letting it send data to sink node (base station) at a very high rate (we used data sending rate of 480 Bytes/second). We have emulated the activity events by generating three independent sequences of active nodes (indicating motion trails) each in one of the three floors. From a remote server, periodic serial message (containing new active node numbers) is sent to the sensor motes in the testbed to generate the activity sequences. The sensor nodes receiving the serial message with it's ID start generating sensor data. Other nodes act as relay only. This periodic activation of nodes through serial message follow the activity transitions defined in the corresponding $ATPG$. In this way the activity transition experiments are performed with network-wide data collection. In addition each node periodically sends one local status data packet (containing information of remaining energy, duty cycle, hop count etc.) to sink every 30 seconds.

Comparison: For performance comparison we have compared combinations of existing routing schemes with selected MAC protocol, with *ActRouting* (*ActiSen* without *ActDutyCycling*) to show performance improvement due to *ActRouting*, and then compared with whole *ActiSen* system (*ActRouting* + *ActDutyCycling* on top of selected MAC protocol) to show performance improvement due to both activity aware routing and duty cycling. Following relevant routing protocols are used: $PMin$ (shortest path routing), CTP [33] (very commonly used data collection protocol for sensor networks, that uses link and path quality), and $CMAX$ (an energy aware protocol [25] where data is forwarded preferably to neighbor with higher remaining energy in the neighborhood). As explained earlier, *ActDutyCycling* in *ActiSen* system doesn't propose a new MAC protocol, but offers performance improvement of the MAC protocol in use by adapting

the effective sleep interval with activity patterns. In the real testbed of $TelosB$ motes we have used the $X-MAC$ [16] as the base MAC protocol. In the experiments we have compared the following configurations:

(a) *PMin + X-MAC*: $PMin + X-MAC$ protocol.
(b) *CTP + X-MAC*: $CTP + X-MAC$ protocol.
(c) *CMAX + X-MAC*: $CMAX + X-MAC$ protocol.
(d) *ActRouting + X-MAC*: $ActRouting + X-MAC$ protocol.
(e) *ActiSen*: $ActRouting + ActDutyCycling$ (using selected $X-MAC$ protocol).

The 82 node network formed a 9 hop routing tree with -5 dBm transmission power of CC2420 radio of TelosB motes. The sink node is in middle of the three floors, as shown in Figure 15(b). In this network data collection scenario we have evaluated following parameters: (i) data delivery latency, (ii) data throughput, (iii) minimum node energy in the network through time (indicating network lifetime), (iv) duty cycle of the nodes, (v) network energy consumption etc. For radio duty cycling X-MAC is used with 5 seconds sleep interval, while *ActDutyCycling* (used on top of X-MAC), uses sleep interval range between TS^{max} = 10 seconds and TS^{min} = 1 second.

4.1.1. Performance Evaluation

Now we describe the performance analysis of our proposed *ActiSen* system compared to existing protocols.

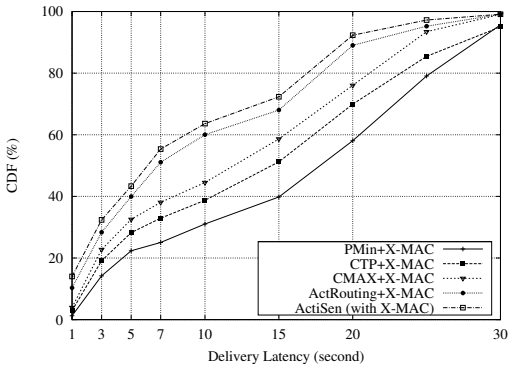


Figure 16: Distribution of data delivery latency

Latency: Figure 16 represents the distribution of delivery latency of packets in the 82 node network. It can be observed that *ActRouting + X-MAC* provides much lower latency than each of the comparing protocols (*PMin + X-MAC*, *CTP + X-MAC*, *CMAX + X-MAC*), and then the whole *ActiSen* system performs even better. In *PMin + X-MAC*, *CTP + X-MAC*, *CMAX + X-MAC*, 70% of the packets are delivered with latency between 20 seconds to 25 seconds. But in *ActRouting + X-MAC* and *ActiSen*, the 70% of the packets are delivered within latency around 13 seconds to 15 seconds. Therefore *ActiSen* provides much lower delivery latency, providing better performance to the application. *ActiSen* achieves this improvement in latency by (a) avoiding selection of currently active nodes (which are busy with sensing and sending own data) as relays by *ActRouting*, and (b) adaptive increase in duty cycle from *ActDutyCycling* for the active and

predicted active nodes. Even in case the activity transitions don't always follow the *ATPG*, reactive duty cycle adjustments in *ActRouting* and *ActDutyCycling* provides improved latency.

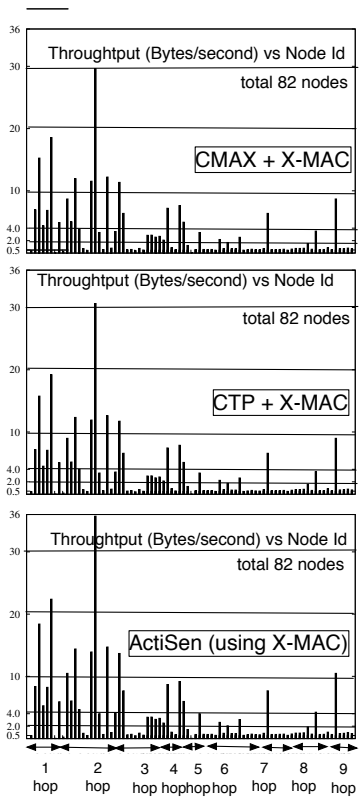


Figure 17: Data throughput at Sink

Data Throughput: Figure 17 shows the data throughput for each node at sink. More throughput indicates more event data successfully delivered and reported at sink. It is observed that for each of the 82 nodes, *ActiSen* provides much improved data throughput than others. For all the 82 nodes *ActiSen* provides a data throughput improvement ranging from 5% to 13%. This advantage in *ActiSen* comes from (a) higher duty cycle on the route containing nodes between active data sources and the sink, and (b) avoiding selection of currently active nodes (which are busy with sensing and sending own data) as relays by *ActRouting*.

Lifetime: Figure 18 represents the minimum energy of any node in the entire network through time. This property decides the network lifetime. The energy consumption is configured in such a way that all nodes start with energy 2200 mAh. For indicating the rate of drop in minimum remaining node energy in network, Figure 18 represents the energy drop with respect to a chosen value of 0.05806 mAh. This value is chosen for purpose of analysis because the minimum node energy in $PMin + X-MAC$ reduces by an amount of nearly 0.05806 mAh energy in experiment run time of 1800 seconds (i.e. 30 minutes). This chosen value for analysis doesn't affect the nature of energy consumption of network. Now it can be observed that the minimum energy of any node in *PMin + X-MAC*, *CTP + X-MAC* and *CMAX + X-MAC* depletes faster

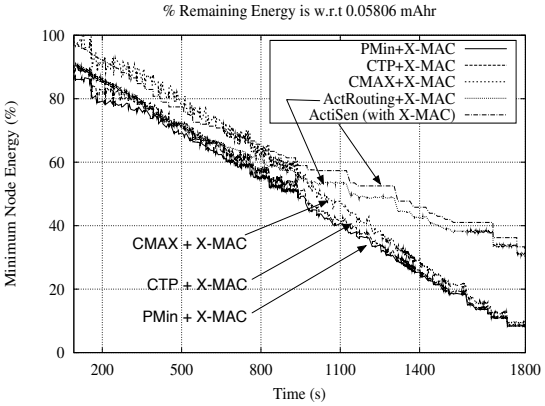


Figure 18: Minimum node energy in network

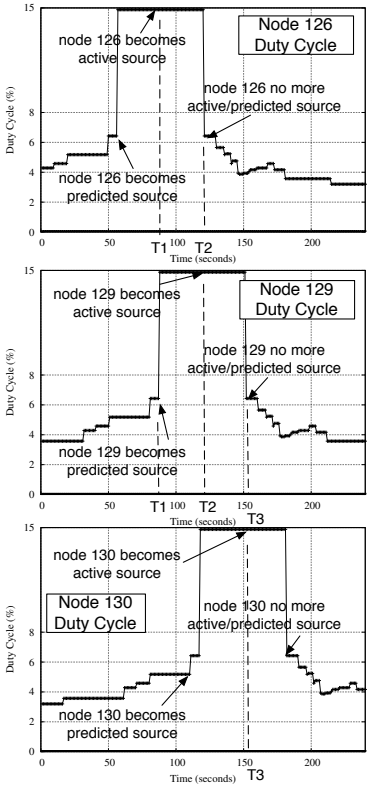


Figure 19: Dynamic radio duty cycle due to *ActDutyCycling*

than the one in *ActiSen*. Within 1800 seconds time, the minimum energy of any node in *PMin + X-MAC*, *CTP + X-MAC* and *CMAX + X-MAC* reduces by an amount close to 0.05806 mAh. Therefore in protocols other than *ActiSen* network node depletes almost all it's energy within time 1800 seconds, had the network start with 0.05806 mAh for all nodes. But in same scenario in same time the minimum energy of any node in *ActiSen* would have still around 33% energy left. Therefore it is clear that network lifetime for *ActiSen* will also be much higher than others. *ActiSen* achieves this advantage because of (i) *ActDutyCycling*'s adaptively lowering of duty cycle during non-active phase of nodes and (ii) *ActRouting*'s energy balanced relay selection.

Dynamic duty cycling: Figure 19 shows an example of dynamic configuration of sleep interval (therefore dynamic change of duty cycle) of a node because of *ActDutyCycling* protocol in *Actisen*. The dynamic duty cycle in *Actisen* is illustrated with the example of activity transition from node 126 to node 129 to node 130 (126 → 129 → 130). During the experiment the node 126 was maintaining a lower effective duty cycle. But according to *ATPG* when predicted source is closer to that node its duty cycle gradually increases because of the information gradient. Then the node becomes the predicted source and increases the duty cycle to around 15%. It continues the duty cycle of 15% since it turns into active source after some time. At the same time node 129 becomes predicted source and it's duty cycle gets higher. But when the node 126 is no more an active, the duty cycle gradually drops to low value. Then node 129 becomes active source and node 130 becomes predicted source. In similar manner as node 126, the duty cycle of nodes 129 and 130 gets updated. This shows an example of duty cycle change during activity transition in the network.

4.2. Effect of Exception in Activity Transition

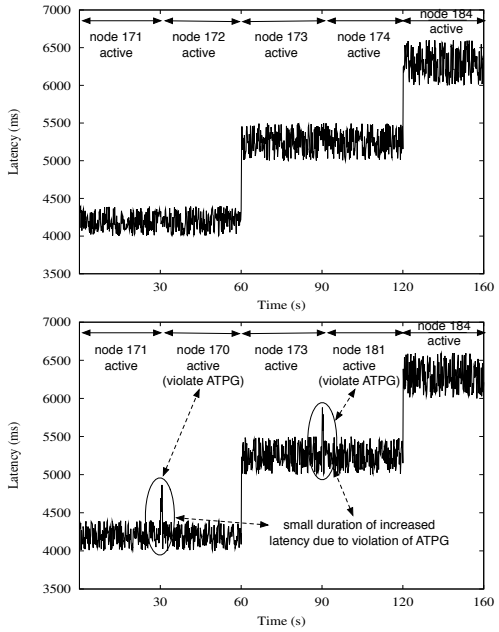


Figure 20: Effect of exception in activity transition predicted by *ATPG*

The design of Activity Transition Probability Graph *ATPG* is fairly accurate in predicting all the probable activity transitions, and it is also adaptive to changing nature in activities. This is also validated with real-world practical observations in *CASAS* Smart Home testbeds [1]. Our activity-aware design of *ActiSen* still have the capability to adapt to situations, where the activity transition has exception and doesn't follow the *ATPG*. Figure 20 shows such a scenario where the *ActiSen* system automatically adapts to exception in *ATPG*. According to *ATPG* one probable activity sequence is → 171 → 172 → 173 → 174 → 184. Each node becomes active source for around 30 seconds. The corresponding data delivery latency for each active source is shown on top of Figure 20. But in the case of exception of *ATPG* where activity transition is → 171 → 170 → 173 → 181 → 184, nodes 170 and 181 becomes active even they were not predicted to be among next active sources. But it can be seen that the delivery latency increases only by small amount and within small time duration, the delivery latency reduces to normal value, same as the latency if the corresponding node was active source according to *ATPG*.

To note that this example partially illustrate the advantage of *ActiSen*, based on the observed fast reactive adjustment. The data source nodes were not far away (in hopcounts) from sink. So the short path from the nodes to sink increased the duty cycle relatively fast. But the advantage of *ActiSen* will be also visible for examples with nodes further away from sink, where it will take some time for the nodes on the routing path to gradually increase their duty cycle and thus reduce delivery latency. Therefore there is a tradeoff between how fast the reactive strategy readjusts the latency and the advantage of reactive strategy in *ActiSen*. If the delay for re-adjustment is too low, it may constrain the advantage due to *ActiSen*, compared to totally reactive strategy. Overall, the capability of *ActiSen* to effectively handle exception activities comes from reactive adjustment of duty cycle in *ActRouting* and reactive gradient update by new active source in *ActDutyCycling*. Therefore the adverse effect of any exception activities is minimized by efficient design of *ActiSen*.

4.3. Experiment in Simulation

In TOSSIM sensor network simulator set up with different network size, the motion activity trajectory is simulated by periodically activating (setting nodes as the data source) a trajectory of nodes one by one with an interval of 30 seconds. In the simulation physical separation between the closest nodes was 5 meter. The radio range was set to the default, and each node had multiple neighbors, more than just the physically closest neighbors. For each network configuration, we have evaluated the following parameters: (i) mean delivery latency, (ii) data throughput at sink, (iii) projected network lifetime, and (iv) network energy consumption.

Evaluation environment: We have evaluated our proposed *ActiSen* system in TinyOS sensor network simulator TOSSIM [34] with varying network size from 20 to 100, to validate the scalability of proposed activity-aware design. For each network size, the nodes are deployed randomly in the area so that the minimum nodes separation is at least 5 meters. The sink node is at one corner of the deployed area. The simulation scenario is set up like a smart environment where detected activity information is reported to the sink.

Activity transition and data generation: In TOSSIM simulation experiments a python program periodcially decides the active nodes and the activity transitions. This controls the order of activie nodes in the network.

Comparison protocols: For performance comparison, following relevant routing protocols are used: *PMin* (shortest path routing), *CTP* [33] (link and path quality aware routing), and

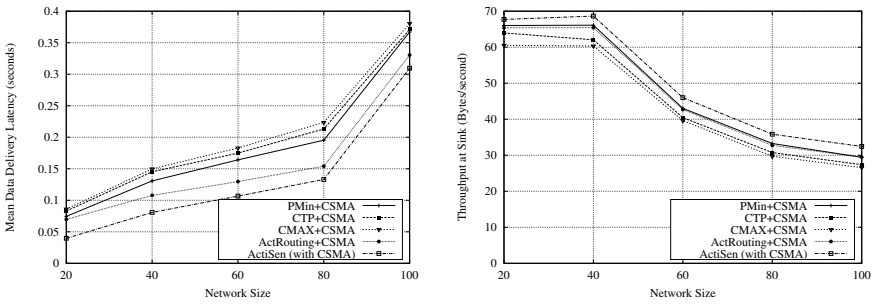
CMAX (energy aware routing). As explained earlier, *ActDutyCycling* in *ActiSen* system doesn't propose a new MAC protocol, but offers performance improvement of the MAC protocol in use by adapting the effective sleep interval. For MAC protocol in TOSSIM simulation we have used CSMA.

In the simulation experiments we have compared the following configurations:
(a) *PMin + CSMA*: *PMin* (shortest path routing) + *CSMA*
(b) *CTP + CSMA*: *CTP + CSMA*
(c) *CMAX + CSMA*: *CMAX* (energy aware routing) + *CSMA*
(d) *ActRouting + CSMA*: *ActRouting + CSMA*
(e) *ActiSen*: *ActRouting + ActDutyCycling* with selected *CSMA* protocol

4.3.1. Performance Evaluation

Data delivery latency: Figure 21(a) shows the mean data delivery latency for the selected data sources for varying network size. Now it is observed that *ActiSen* has much lower average latency than other for all network size. The improvement in latency ranges from 15% upto 46%. This shows that *ActiSen* provides better application performance with reduced latency in event data delivery and notification, which is also scalable from small to large network size.

Data throughput: Figure 21(b) shows the overall data throughput at sink for varying network size. It is observed that *ActiSen* has much higher data throughput than others for all network size. The improvement in throughput ranges from 2% upto 11%. This shows that *ActiSen* provides better application performance with improved data throughput in event data delivery and notification, which is also scalable from small to large network size.



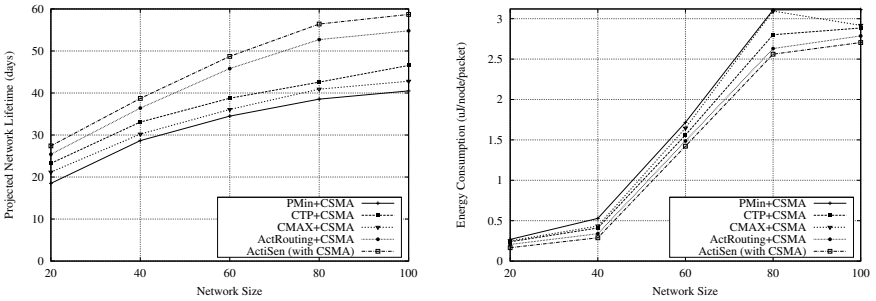
(a) (b)

Figure 21: (a) Mean data delivery latency (seconds) with varying network size (b) Data throughput at Sink (Base Station) with varying network size

Lifetime: Figure 22(a) shows that the network lifetime of network using *ActiSen* is significantly higher than that of others for all network size. *ActiSen* achieves improvement in network lifetime ranging from 17.1% upto 48.2%. The *ActRouting* and *ActDutyCycling* both contribute to this advantage. *ActDutyCycling* reduces the energy consumption of nodes when they are outside active region. *ActRouting* enables energy efficient selection of relay nodes on the active route. These lead to more balance in energy consumption of nodes in network. These all finally lead to effective decrease in node energy consumption in the network, resulting in improved lifetime.

Network energy consumption: The improvement in duty cycle ensures reduced energy consumption for *ActiSen* for the network. This is shown in Figure 22(b) which shows average

node energy consumption per successfully delivered packet at sink. This is a representative of network energy consumption which in turn denotes energy efficiency. It is observed that the average network energy consumption is much lower in *ActiSen* than others, and it is scalable with network size. This property of *ActiSen* gives the advantage of resource optimization for resource constrained sensor networks.



(a) (b)

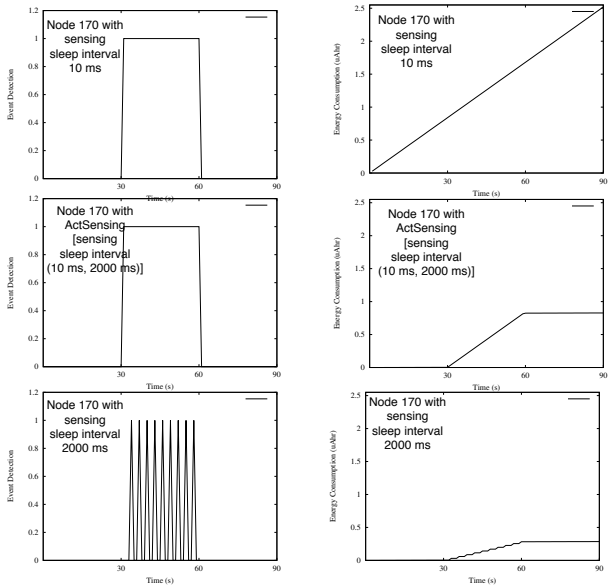
Figure 22: (a) Projected network lifetime (days) with varying network size (b) Energy consumption per successfully delivered message per node (uJ/packet/node) with varying network size

4.4. Performance Evaluation of Activity-Aware Sensing

Here we have illustrated the advantage of proposed activity-aware sensing method called *ActSensing*. Figure 23(a) and 23(b) show the event detection accuracy and sensing energy consumption by node 170 for the first 90 seconds time interval of the event generation as in Figure 20, with exception event where 170 becomes active against the prediction in *ATPG*. They show that *ActSensing* (here the range for adaptive sleep interval is configured as (10 ms, 2000 ms)) gives accurate event detection even in the exception events, but also saves sensing energy consumption for sensor node. Figure 23(a) shows that despite not predicted to be active source, sensibly chosen lower bound for sensing sleep interval in *ActSensing* doesn't affect the event detection accuracy, even if node is not predicted to be active. Figure 23(b) shows that *ActSensing* can save significant sensing energy. The sensing energy spent, depends on the sensor being used. We have used the sensing energy for Panasonic AMN-31111 PIR motion sensor [35], that we have used in our own motion sensor network testbed (the evaluation Harvard Motelab testbed doesn't contain motion sensor). So it is clear that the using fixed sleep interval can either degrade event detection accuracy or can lead to much higher energy consumption, compared to *ActSensing*.

5. Conclusion

This paper presents *ActiSen*, an activity-aware system design for wireless sensor networks, that can learn from detected activity patterns. The activity-aware design creates and updates an Activity Transition Probability Graph (ATPG) and then retroactively utilizes knowledge from it to dynamically configure the sensing, routing and radio duty cycling operations for providing: better performance to application, and resource optimization for resource constrained system. The activity-aware design achieves this using activity-aware sensing, activity-aware radio duty cycling, and activity-aware energy-balanced routing. The experimental results from real testbed



(a) (b)

Figure 23: (a) Event detection accuracy (b) Energy consumption for sensing

and simulation experiments validate the advantages of activity-aware design and also show its scalability with varying network size.

6. Acknowledgement

This work is done as a part of the project *Activity-Aware Sensor Network for Smart Environments* (http://sensorweb.cs.gsu.edu/research/actisen.html). This is supported by NSF NetSE program under the research grant CNS 0914371.

References

[1] CASAS Smart Home Project, http://ailab.wsu.edu/casas/.
[2] MavHome, http://ailab.uta.edu/mavhome/.
[3] D. Lymberopoulos, T. Teixeira, A. Savvides, The BehaviorScope Framework for Enabling Ambient Assisted Living, Special Issue of International Journal on Personal and Ubiquitous Computing (to appear).
[4] A. Wood, J. Stankovic, G. Virone, L. Selavo, Z. He, Q. Cao, T. Doan, Y. Wu, L. Fang, R. Stoleru, Context-aware wireless sensor networks for assisted living and residential monitoring, Special Issue IEEE Networks 22 (4) (2008) 26–33.
[5] Smart Home-based Health Platform for Behavioral Monitoring and Alteration for Diabetic and Obese Individuals, http://www.icta.ufl.edu/projects_nih.htm.
[6] Smart Medical Home, http://www.futurehealth.rochester.edu/smart_home/.
[7] Spinner, http://www.media.mit.edu/resenv/spinner/introduction.html.
[8] V. Srinivasan, J. Stankovic, KaminWhitehouse, Protecting your Daily In-Home Activity Information from a Wireless Snooping Attack, in: UbiComp, 2008.
[9] M. Buettner, R. Prasad, M. Philipose, DavidWetherall, Recognizing Daily Activities with RFID-Based Sensors, in: UbiComp, 2009.

[10] Smart Environments, http://wsnl.stanford.edu/smartenv.html.
[11] S. Ahn, D. Kim, Proactive Context-Aware Sensor Networks, in: EWSN, 2006.
[12] R. Menchaca-Mendez, J. J. Garcia-Luna-Aceves, Robust and Scalable Integrated Routing inMANETs Using Context-Aware Ordered Meshes, in: INFOCOM, 2010.
[13] W. Ye, J. Heidemann, D. Estrin, An energy-efficient MAC protocol for wireless sensor networks, in: 21st Conference of the IEEE Computer and Communications Societies (INFOCOM), 2002.
[14] T. van Dam, K. Langendoen, An Adaptive Energy-Efficient MAC Protocol for Wireless Sensor Networks, in: The 1st ACM Conference on Embedded Networked Sensor Systems (SenSys), 2003.
[15] J. Polastre, J. Hill, D. Culler, Versatile Low Power Media Access for Wireless Sensor Networks, in: SenSys, 2004.
[16] M. Buettner, G. V. Yee, E. Anderson, R. Han, X-MAC: a short preamble MAC protocol for duty-cycled wireless sensor networks, 2006.
[17] H. Lin, M. Lu, N. Milosavljevic, J. Gao, Composable Information Gradients in Wireless Sensor Networks, in: IPSN 2008, 2008.
[18] B. Kusy, H. Lee, M. Wicke, N. Milosavljevic, L. Guibas, Predictive QoS Routing to Mobile Sinks in Wireless Sensor Networks, in: IPSN 2009, 2009.
[19] P. Huang, H. Chen, G. Xing, Y. Tan, SGF: A state-free gradient-based forwarding protocol for wireless sensor networks, ACM Transactions on Sensor Networks (TOSN) 5 (2).
[20] S. Tang, D. De, W.-Z. Song, D. Cook, S. Das, ActSee: Activity-Aware Radio Duty-Cycling for Sensor Networks in Smart Environments, in: Eighth IEEE International Conference on Networked Sensing Systems (IEEE INSS), 2011.
[21] J. H. Chang, L. Tassiulas, Energy Conserving Routing in Wireless Ad-hoc Networks, in: INFOCOM, 2000, pp. 22–31.
[22] J. H. Chang, L. Tassiulas, Fast Approximate Algorithms for Maximum Lifetime Routing in Wireless Ad-hoc Networks, in: Networking, 2000.
[23] Q. Li, J. Aslam, D. Rus, Online power-aware routing in wireless Ad-hoc networks, in: Mobicom, 2001.
[24] Y. Xue, Y. Cui, K. Nahrstedt, A Utility-based Distributed Maximum Lifetime Routing Algorithm forWirelessNetworks, in: QShine, 2005.
[25] K. Kar, M. Kodialam, T. V. Lakshman, L. Tassiulas, R. Tassiulas, Routing for Network Capacity Maximization in Energy-constrained Ad-hoc Networks, in: INFOCOM, 2003.
[26] L. Lin, N. B. Shroff, R. Srikant, Asymptotically Optimal Energy-Aware Routing for Multihop Wireless Networks With Renewable Energy Sources, IEEE/ACM Transactions on Networking 15.
[27] C. Ok, P. Mitra, S. Lee, S. Kumara, Maximum Energy Welfare Routing in Wireless Sensor Networks, in: NETWORKING 2007, 2007.
[28] C. Sengul, R. Kravets, Conserving Energy with On-Demand Topology Management, in: MASS 2005.
[29] G. Lu, D. De, M. Xu, W.-Z. Song, B. Shirazi, TelosW: Enabling Ultra-Low Power Wake-On Sensor Network, in: INSS 2010, 2010.
[30] A. B. Atkinson, On the measurement of inequality, Journal of Economics Theory.
[31] G. Werner-Allen, P. Swieskowski, M. Welsh, MoteLab: a wireless sensor network testbed, in: Fourth International Symposium on Information Processing in Sensor Networks (IPSN), 2005.
[32] Y. Peng, W-Z. Song, R. Huang, M. Xu, B. Shirazi, R. LaHusen, G. Pei, Cacades: A Reliable Dissemination Protocol for Data Collection Sensor Network, in: IEEE Aerospace Conference, 2009.
[33] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, P. Levis, Collection Tree Protocol, in: Proc. of the 7th ACM Conference on Embedded Networked Sensor Systems (SenSys), 2009.
[34] N. L. Philip Levis, TOSSIM: A Simulator for TinyOS Networks (Jun. 2003).
[35] Panasonic Motion Sensor (Passive Infrared Type), http://pewa.panasonic.com/pcsd/product/sens/pdf/amn.pdf.