Collaborative Two-Level Task Scheduling for Wireless Sensor Nodes with Multiple Sensing Units

H. Ozgur Sanli, Rajesh Poornachandran and Hasan Çam
Department of Computer Science and Engineering
Arizona State University
Tempe, AZ 85287
Email: {hedo, rajesh.p, hasan.cam}@asu.edu

***Abstract*— A sensor node with multiple sensing units is usually unable to process simultaneously the data generated by multiple sensing units, thereby resulting in event misses. This paper presents a collaborative task scheduling algorithm, called CTAS, to minimize event misses and energy consumption by exploiting power modes and overlapping sensing areas of sensor nodes. The novel idea of CTAS lies in that it employs a two-level scheduling approach to the execution of tasks collaboratively at group and individual levels among neighboring sensor nodes. CTAS first implements coarse-grain scheduling at the group level to schedule the event types to be detected by each group member. Then, CTAS performs fine-grain scheduling to schedule the tasks corresponding to the assigned event types. The coarse grain scheduling of CTAS is based on a new algorithm that determines the degree of overlapping among neighboring sensor nodes. Simulation results show that CTAS yields significant improvements in energy consumption up to 67% and reduction in event misses by 75%.**

I. Introduction

Wireless sensor networks emerged as an important wireless technology with the advances in sensor architectures such as the inclusion of multiple sensing units and other components with variable power mode capability. The sensor nodes exploit the availability of multiple power modes by selecting low power modes when they are idle. While saving energy, this also introduces the problem of data accuracy due to the latency involved in switching a sensor node from an energy saving low power mode to the high power mode required for event processing. This latency may be long enough to cause the sensor node to miss the processing of an event on time. Fortunately, the accuracy of data can be improved by making use of the fact that multiple sensor nodes observe the same physical region in densely deployed sensor networks. However, for such an approach to minimize energy consumption and event misses, we address how the tasks corresponding to the events of interest can be executed collaboratively among a group of neighboring sensor nodes with multiple sensing units. In this paper, we use the term ”task” to refer to the required processing of data generated by a sensing unit upon occurrence of an event.

We introduce a *C*ollaborative two-level *Ta*sk *S*cheduling algorithm, called CTAS, for wireless sensor nodes with multiple sensing units. CTAS employs both coarse-grain scheduling and fine-grain scheduling. In coarse-grain scheduling at group level, CTAS schedules the event types and data transmissions for a group of neighboring sensor nodes. The coarse-grain scheduling is based on the degree of their overlapping sensing areas determined by a proposed coverage testing scheme. The scheduling of event-types and data transmission in CTAS are referred to as *event-types scheduling* and *data transmission scheduling* respectively. In fine-grain scheduling at individual node level, CTAS schedules the tasks of the event types assigned by the coarse-grain scheduling at each sensor node. When a sensor node of a group is not assigned a particular event type, the sensor node shuts down the sensing unit corresponding to that event type until the next assignment phase of coarse-grain scheduling. The contributions of CTAS are as follows.

- CTAS enables sensor nodes to keep only a subset of their sensing units active at any time even though each sensor node has multiple sensing units. While some of the sensing tasks are scheduled to neighboring nodes, the coverage degree of the network is still maintained at a specific level for each event type.
- In comparison with the existing techniques utilizing complete overlaps among sensing regions, CTAS can also help sensor nodes which only have partial overlapping sensing regions with their neighboring nodes. This is achieved by having sensor nodes periodically switch their active sensing units for the assigned event types to sleep mode without missing the maximum signal strength for each event occurrence. In this process, the degree of overlap between the active times of sensing units at neighboring nodes is determined according to the reliability requirements of the corresponding event type. In addition, if multiple sensor nodes observe the same event, only the selected sensor node(s) transmit data for the event.
- CTAS provides realtime and reliable data collection by having sensor nodes send short control packets after each event occurrence. The cluster-heads select the suitable sensor nodes based on control packets according to the priority and reliability requirements of events.
- We propose a new coverage testing method to calculate the covered percentage of a node’s sensing region by its neighboring nodes’ sensing regions. This method is

[1]This research is supported in part by the CEINT grant CRS0044.

used to determine the type of task scheduling (sensing / data transmission) for each one of the node's sensing units. The existing methods in the literature consider only the neighboring nodes that are located within the sensing region of the node for coverage testing. In the proposed approach, we consider both the nodes that reside outside the sensing range of the node as well as the ones that are located inside. Therefore, the proposed coverage testing method is more accurate in determining the covered percentage of a node's sensing region.

The rest of the paper is organized as follows. Section II describes the related work. Section III explains the system model and assumptions. Section IV gives the description of CTAS. Simulation results are presented in Section V. Concluding remarks are made in Section VI.

II. Related Work

The scheduling of radio operation has been a popular problem in wireless networks since communication constitutes a major source of energy consumption of the nodes. The work in [1] presents a distributed, topology-based algorithm during which each sensor node becomes a coordinator if its neighboring nodes cannot communicate directly or with the help of coordinator nodes. The communication backbone construction algorithm of [2] identifies equivalent nodes in terms of routing perspective with the help of a virtual grid of squares such that only one sensor node keeps its radio active for each square. The work in [3] puts sensor nodes into sleep mode which wake up periodically to retrieve their packets stored at neighboring nodes.

The coverage preserving node scheduling algorithms utilize the dense network deployment to reduce the number of nodes with active sensing units. The authors of [4] introduce the concept of "sponsored coverage" to first identify nodes whose sensing regions are covered by their neighboring nodes and then to deactivate the sensing units of such nodes. The work in [5] presents coverage service protocols which can provide multi-level coverage support over the target region with the help of a virtual grid structure imposed on the target region. The algorithm proposed in [6] does not require location information of sensor nodes for coverage scheduling, however, their solution is probabilistic and can not guarantee complete coverage of the target region. The main drawback of these efforts is that they are not applicable when the sensing region of a node is only partially covered by its neighboring nodes. In this paper, we also make use of partial overlaps in nodes' sensing regions in addition to the complete overlaps to avoid redundant data generation and transmission. In [7], differentiated surveillance of the target region is achieved by checking the coverage of a discrete set of points. The problem with their approach is the overhead of coverage computation which requires storage / update of information for each grid point.

The wireless sensor network should be able to transfer data to the base station in realtime. In particular for surveillance sensor networks such as border monitoring networks, the

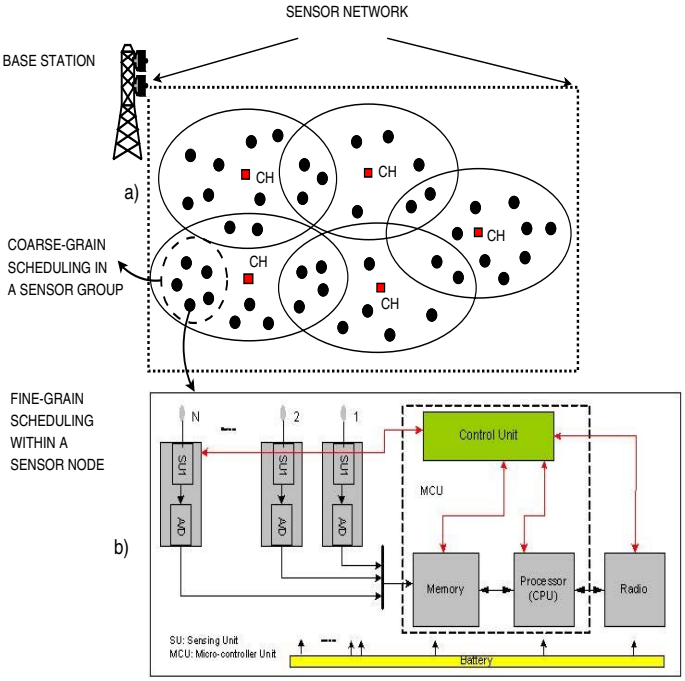

Fig. 1. a) System Model b) Sensor Node Architecture

preemptive steps can be taken against the intruder on time if the data is delivered to the base station under deadline constraints. However, it is a challenging problem to provide QoS (Quality of Service) in the resource constrained sensor network environment. The work in [8] introduces a realtime communication architecture in which a packet's priority is determined both based on its deadline and its distance to the destination. The same approach is used in [9] to assign packet priorities which also presents a prioritized medium access layer to give preference to realtime data over non-realtime data. In [10], the authors consider the realtime delivery requirement for sensor data based on its spatio-temporal correlation property. The authors of [11] formulate the problem of realtime data collection over the tree based communication structure of sensor networks with overall energy consumption as the goal, however, their work is not scalable for sensor networks with a large number of nodes. In [12], a cost is assigned for each link in the data gathering tree first and then a K least cost path algorithm is used to find a set of routes which both meet the delay constraints and provide the maximum throughput for non-realtime traffic. In comparison with these efforts which focus only on the delivery aspect of realtime data, CTAS has the advantage of employing a combined approach which incorporates realtime event capturing, realtime data transfer and data reliability.

The percentage of missed events determines the QoS supported by a sensor network, therefore, several task scheduling algorithms have been developed to resolve the conflict between energy savings and event misses. In [13], an algorithm is proposed to address the event misses and task priority re-

quirements while scheduling the tasks within a sensor node in an energy efficient manner. A system-level power management technique is presented in [14] for distributed wireless sensor networks, which makes use of an adaptive shutdown policy by taking event misses into account. The work in [15] introduces a preemptive and priority based scheduling algorithm to allocate sensor resources to the data gathering queries issued. However, these works have the drawback that they consider task assignment at each node separately from its neighboring nodes. Moreover, some of these approaches [14], [15] assume the prior knowledge of the tasks which is not a practical assumption since the occurrences of events may not be deterministic in a sensor network.

III. System Model

We consider cluster-based wireless sensor networks in which data is first aggregated at cluster-heads and then transferred to the base station. Figure 1 shows a higher level of view of the network model used in this paper. The sensor network consists of a large number of sensor nodes in which nodes close to each other with possibly overlapping sensing regions form the sensor groups. Each node has a unique identifier and has its location information either based on a low power GPS (Global Positioning System) receiver [16] or localization methods devised for sensor and ad hoc networks [17]. Sensor nodes are synchronized using GPS or other methods such as time-stamp exchange [3]. The time is divided into discrete slots of duration W that is common to all sensor nodes for the coordination of the sensing operation. The sensor nodes advertise their locations and residual energy levels periodically such that each node has the recent information on the status of its local neighborhood. This is because some sensor nodes may run out of battery earlier than others. The sensor network operates in two layers in which coarse grain scheduling of sensing tasks within node groups is performed first followed by the fine grain scheduling of the event processing tasks within each sensor node.

We use the MICA-2 [18] mote as the hardware model in which the resources like CPU, memory and RF unit are shared for processing and transmitting data from the sensing units as shown in Figure 1. The data flow among the sensor node resources is coordinated by the control unit. The components of a sensor node can operate in multiple power modes and the current power modes of the components determine the state of the sensor node. If a sensor node has κ components, each having S_i power modes, the node can theoretically have as many as $\Pi_{i=1}^{\kappa} S_i$ states. However, all these power modes are not used in real sensor nodes because of the co-activation requirements of their components during data processing [19]. Table I shows the eight valid power states for the MICA-2 mote listed in the descending order of their power consumption. We assume that all sensor nodes are identical and have the same number, types of sensing units denoted by the set M. A sensor node observes an event, which is characterized with its type and duration, only if the event occurs within the sensing range r_s of the corresponding sensing unit and if the strength of the signal generated by the event is above or below a particular threshold. In this paper, timers are used to give priority to sensor nodes on their channel access for message broadcasts. We note that this priority mechanism can also be implemented by having the cluster-heads coordinate the medium access of their members.

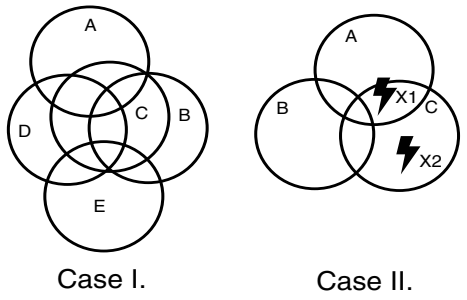

Fig. 2. The sensor node C has its sensing range covered almost completely in Case I. The sensor nodes whose sensing ranges partially overlap are illustrated in Case II where X_1 and X_2 are two different events.

IV. Collaborative Two Level Task Scheduling

A. Algorithm Overview

The operation of CTAS is explained using the group of nodes shown in Figure 2. In Case I, node C's sensing region completely overlaps with its neighboring nodes namely A, B, D and E. It is important to point out that the figure only shows a local neighborhood in the sensor network where each sensor node may be overlapping with other sensor nodes that are not shown in the figure. If node C is in critical condition in terms of available battery power and turns off its sensing unit, it can offload the sensing tasks corresponding to this event type to its neighboring nodes due to the complete coverage of its sensing region. However, when the sensing region of node C partially overlaps with the sensing regions of nodes A and B as shown in Case II, only node C can transmit packets to the cluster-head for events from its non-overlapping region (X2 in Figure 2). At the same time, node C should be able to offload the data communication associated with events occurring in the overlapping region to node A (X1 in Figure 2). The coarse-grain scheduling phase of the algorithm addresses both cases with *event-type scheduling* and *data transmission scheduling* depending on the coverage degree of the node's sensing region by its neighboring nodes as follows.

- CTAS performs event type scheduling for the almost complete overlap case shown in Figure 2. In this case, node C ceases sensing certain event types for a fixed period based on the trust that its neighboring nodes A, B, D and E are going to capture the events occurring in its sensing region. In the algorithm, a verification procedure is included to avoid any one of these neighboring nodes to turn off their sensing units at the same time with node C. Then, sensor nodes keep their sensing units active only for those event types that are assigned to them at coarse-grain scheduling phase. The processing of events from

TABLE I

THE SET OF VALID POWER STATES FOR A SENSOR NODE ARRANGED IN DECREASING ORDER OF POWER CONSUMPTION.

Power State of Sensor Node	*Status of Sensing Units*	*Radio*	*Processor*	*Status of Memory Units*
$S1$	At least one sensing unit is active	Tr-Rx	Active	Instruction or data memory is active
$S2$	At least one sensing unit is active	Rx	Idle	Instruction and data memory are off
$S3$	At least one sensing unit is active	Rx	Sleep	Instruction and data memory are off
$S4$	At least one sensing unit is active	Off	Sleep	Instruction and data memory are off
$S5$	All sensing units are off	Tr-Rx	Active	Instruction or data memory is active
$S6$	All sensing units are off	Rx	Idle	Instruction and data memory are off
$S7$	All sensing units are off	Rx	Sleep	Instruction and data memory are off
$S8$	All sensing units are off	Off	Sleep	Instruction and data memory are off

the active sensing units are coordinated with fine-grain scheduling.

- The data transmission scheduling is applied for the partial overlap case in Figure 2. We note that in such cases, a sensor node can not assure that each event occurring in its sensing region will be observed by its neighboring nodes if its sensing unit is deactivated. Thus, when an event occurs, the sensor node waits for a duration proportional to the ratio of its residual energy to the initial energy level in a short time window. The sensor node then transmits a small-sized control packet informing the cluster-head on the type of the event and its priority requirements. Finally, the cluster-head determines which sensor node(s) should transmit data containing detailed information for the event. The remaining sensor nodes which observed the same event cancel their data transmission for this event and save energy.

The scheduling of tasks at the group level based on the coverage based relationship of sensor nodes is a promising idea based on the following observations. First, if the region covered by a sensor node is being observed by its neighboring nodes, the data sent by the sensor node becomes redundant. This is because the neighboring nodes are already transmitting information for the common events. Such sensor nodes not only save energy based on the elimination of redundant data transmission activity but also from their deactivated sensing units. Second, only the selected sensor nodes broadcast a data packet for each event occurrence while others suppress their broadcast. The neighboring nodes do not transmit the whole packet for an event but only a small sized representative information using control packets to the cluster-head. Third, the redundancy can only be eliminated once the data reaches the cluster-head which is the common point for neighboring sensor nodes. Therefore, sensor node to cluster-head transmission is inevitable in the data aggregation process. In our approach, the efficiency of data aggregation is greatly enhanced since sensor nodes do not generate redundant data in the first place which avoids transmission of redundant data packets to the cluster-head. Each event is often sensed by multiple sensor nodes in dense sensor networks, therefore, energy consumption of sensor nodes is significantly improved with the application of CTAS. Finally, the cluster-heads are selected from sensor nodes and have the same resources as compared to the cluster members. Since the cluster-heads already spend more energy for implementing data aggregation, it is crucial to reduce the data reception load on these nodes. The use of short control packets instead of data packets to eliminate data redundancy helps cluster-heads save energy and extends the lifetime of the sensor network.

B. Algorithm Description

The algorithm operates in two layers and follows a round based execution model in which the network lifetime is divided into rounds of fixed period T $(T = kW,\ k \gg 1)$. In the higher layer, at the first slot of each round, event-type scheduling is done for sensing regions that are almost completely covered by neighboring nodes' sensing regions. The "complete coverage" case is defined with respect to the coverage threshold value ρ that is given as input to the algorithm. This parameter enables the user to trade off coverage QoS in a sensor network with the energy consumption. For those sensing units with only partial coverage, data transmission tasks are scheduled among the sensor node group in the rest of the round according to the residual energy of nodes, timing and reliability requirements of the events. In the lower layer, the sensor nodes apply fine-grain scheduling to handle the processing of event tasks since the sensing units share the same limited resources of the sensor node. In addition, the sensor node should be in a certain power state while processing events of a particular type, otherwise, the time overhead of bringing the sensor node from its current power state to the required one may cause the sensor node to miss the event. Therefore, the fine-grain scheduling phase also selects the best power mode for the sensor nodes to reduce the event misses.

The coarse-grain scheduling phase of the algorithm needs prerequisite information on the coverage based relationship of neighboring sensor nodes in a group. In the following section, we first give the definition of the *CoverageTest* function that is used to calculate the covered portion of a node's sensing region by its neighboring nodes.

C. The Coverage Degree of a Node's Sensing Region by Its Neighboring Nodes

The *CoverageTest* function performs coverage calculation by taking the intersection of a node's sensing region with its neighboring nodes' sensing regions. The previous approaches in coverage testing for sensing regions [4], [5], [6] can only consider neighboring nodes that are located *within* the sensing region of the node for which the test is being performed. However, even if the neighboring node is located outside the

FUNCTION 1 *CoverageTest*

Require: N : List of neighboring sensor nodes that have overlapping sensing regions with the node
r_s : Sensing range for the sensing unit
Ensure: The covered portion of the node's sensing region by its neighboring nodes is calculated

1: $(x,y) \leftarrow$ Sensor node location
2: $A_i \leftarrow \emptyset, \ \ i = 1..NR$ *// Set of coverage angles for each ring*
3: $coverageratio \leftarrow 0$ *// The portion of the sensing region covered*
4: **for all** neighboring node **in** N **do**
5: $(x_n, y_n) \leftarrow$ location of the neighboring node
6: *// Determine the contribution of the neighboring node's sensing region to the coverage of each ring*
7: $lowestring \leftarrow$ the lowest id ring whose inner circle intersects with the neighboring node's sensing region
8: **for all** i = $lowestring$ .. NR **do**
9: $< (x_1,y_1),(x_2,y_2) > \leftarrow$ the intersection points of the disks $((x_n,y_n),r_s)$ and $((x,y),r_c*(i-1))$
10: $\alpha \leftarrow \angle(x_1,y_1)(x,y)(x_2,y_2)$
11: $A_i \leftarrow A_i \cup \alpha$
12: **end for**
13: **end for**
14: *// Calculate the coverage degree of each ring and its contribution to overall coverage of sensing region*
15: **for all** i = 1 .. NR **do**
16: $coverageratio \leftarrow \left(\frac{A_i}{2\pi}\frac{\pi r_c^2(i^2-(i-1)^2)}{\pi r_s^2}\right) + coverageratio$
17: **end for**
18: **return** $coverageratio$

sensing region, it can still contribute to the total coverage of the sensing region of the node under consideration as shown in Figure 3. The coverage test depends on the representation of the sensing region of the node as a collection of rings of equal width r_c where $r_s = NR * r_c$ and NR is an integer referring to the total number of rings. The rings are numbered in increasing order starting from the center of the sensing region towards its boundary as illustrated in Figure 3. The accuracy of the ring based approximation of the sensing region improves with increasing NR at the cost of higher computational and storage overhead. The basic idea behind the test is to first check the coverage degree for each ring and then to determine the overall coverage degree by considering the ratio of the rings' areas to the overall sensing region area. The covered portions of each ring are represented with angles for reducing the storage overhead. For each neighboring node, the sensor node determines the ring with the lowest id such that the sensing region of the neighboring node intersects with the inner boundary of the ring closer to the center. As a special case in this process, the neighboring node's sensing region should contain the center of the node under consideration in order to contribute to the coverage of the first ring. Then, starting from the lowest id ring, the portion of each ring within the coverage of this neighboring node is defined with an angle. This angle is the one between the line segments connecting the center of the covered node's sensing region to the intersection points of the ring and the neighboring node's sensing region. Figure 3 also illustrates the angle representing the covered portion of the lowest ring level that is the ring level 5. The function determines the ratio of the union of the angles representing the covered portions of each ring to 2π.

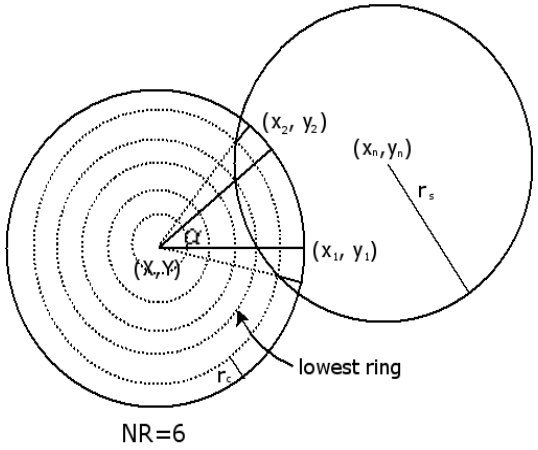

Fig. 3. The covered portion of each ring is represented with an angle

Finally, the total percentage of the sensing region covered is calculated by considering the ratio of the area of each ring to the node's total sensing area in addition to the coverage degree of the ring.

The coverage test described is based on the assumption of *minimum coverage* requirement where it is desirable for a point in the target region to be observed by the least number of active sensing units. The minimum coverage constraint aims to reduce redundant data generation. However, sensor measurements of individual nodes may not be reliable. Therefore, sensor data from neighboring nodes may be combined to achieve data reliability via distributed detection. Such applications impose the *strong coverage* constraint which requires any

event to be detected by at least a given number of sensor nodes. It is straightforward to extend the *CoverageTest* function for the strong coverage requirement. If any area in the target region should be monitored by k sensor nodes at any time, the sensor node should verify that each ring in its sensing region should be covered by the sensing regions of at least k disjoint sets of neighboring sensor nodes.

D. Coarse Grain Scheduling

The sensor nodes use different message types for event-type scheduling and data transmission scheduling. We first describe the contents for each message type and then explain how transmissions of these messages are coordinated within a group of sensor nodes.

- *Sensing Coordination Message (SCM)*: This message is used in event-type scheduling and broadcast by sensor nodes at the beginning of each round only. The SCM message includes the identifier for the sender node, the *Task Delegation Mask* (TDM) and the index of the slot the sensor node starts its sensing operation for each event type that it remains active in the current round. The TDM is a bit mask with number of bits equal to the number of sensing units with each bit for an event type set to one if the node intents to turn off its sensing unit for the corresponding event type. If the transmission range of the nodes is larger than two times their sensing range, the sensor nodes with overlapping sensing regions can receive the SCM messages of each other. Otherwise, the cluster-head has to forward the SCM messages to enable such nodes to receive the necessary information for event-type scheduling.
- *Sensing Information Message (SIM)*: The sensor nodes broadcast these messages after each event occurrence to schedule the transmission of data for the event. The SIM message consists of the following fields in addition to the sender node identifier.
  - *Event type* : This field indicates which sensing unit has detected the event occurrence. In comparison with the TDM which is of length M bits for M sensing units, even type field is $\lceil \log_2 M \rceil$ bits long since each SIM describes the occurrence of a single event.
  - *Received sensing signal strength* : The same event can be observed by multiple sensor nodes, however, their proximity to the event location may vary. The sensor nodes with higher sensing signal strength provide more accurate data for the event.
  - *Event priority* : The granularity of priorities and their assignment to events is determined according to the application requirements.

1) Event-Type Scheduling: The coordination of sensing operation among a group of sensor nodes is performed in the first slot of each round which is referred as the *Consensus Window*. In the Consensus Window, each sensor node agrees with its neighboring nodes on the event types that it should observe by advertising a single SCM. Each sensor node first identifies the event types that contributed most to its overall energy consumption in the previous round. Then, sensor nodes verify that their sensing regions are covered by their neighboring nodes for the selected event types before advertising their intention to offload these event types. Thus, each node updates its information on which neighboring nodes will be active for each event type with the reception of an SCM message. In particular, the sensor node removes the identifier of an SCM sender from its list of active neighboring nodes for each event type with its bit is set to one in the TDM of the message. It is important to note that an SCM also indicates the event types that the broadcasting node will continue to sense in the rest of the round. This information is needed by sensor nodes which have advertised their decision earlier to cease sensing for some event types. Such sensor nodes have to make sure that their neighboring nodes that were assumed to stay active for the corresponding event type will indeed continue their sensing operation in the rest of the round. If at the end of Consensus Window, the sensor node notices that the coverage condition is violated due to coordination errors and changes in the decisions of some of its neighboring nodes, it activates the corresponding sensing unit for the rest of the round to prevent any coverage breaches.

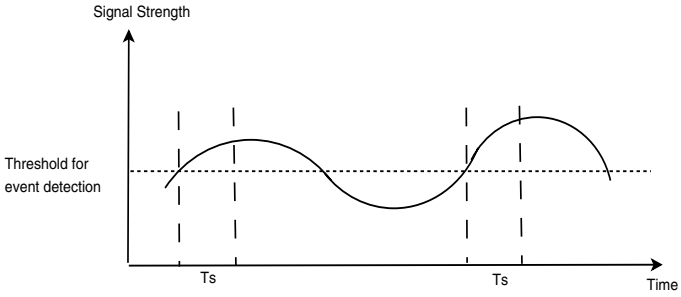

Fig. 4. The sleep mode duration T_s for the sensing unit of an assigned event type is determined according to the characteristics of the observed phenomena

The broadcast of the node's SCM message is scheduled according to the ratio of its residual energy level E_{res} to its initial energy level E_0 using a timer. This serializes the advertisements of nodes in their local neighborhood and gives priority to sensor nodes with less residual energy to turn off their sensing units first. Each node resumes counting down when the channel becomes free. The node identifiers are used to break the tie for neighboring sensor nodes with the same residual energy.

CTAS also employs the following operational approach for improving the lifetime of sensor nodes that have to remain active most of the time due to the topology of their neighboring nodes. In this approach, even if a sensor node remains active for an event type, the sensing unit of the node is periodically switched between active and sleep modes of operation. The periods for active and sleep modes are determined according to the characteristics of event occurrences. The duration of active mode, T_a, is defined as the time required to process an instance of the corresponding event type by sensor nodes.

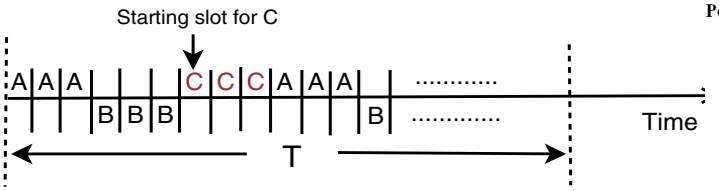

Fig. 5. The sensor node C selects the starting slot that results in the least amount of overlap with the union of used slots for the minimum coverage case ($T_a = 3,\ T_s = 6$)

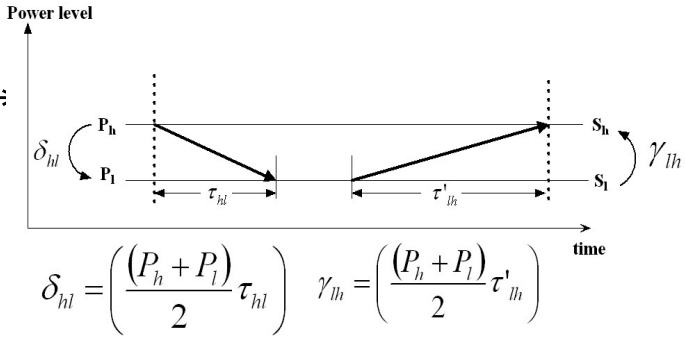

Fig. 6. The energy overhead of transitioning from a less active state S_l to a more active state S_h is denoted by γ_{lh} which is calculated based on the transition duration τ'_{lh} and the power consumption of the states involved in transition ($P_l,\ P_h$). The switching time from S_h to S_l is referred as τ_{hl} which also controls the energy spent (δ_{hl}) for this transition

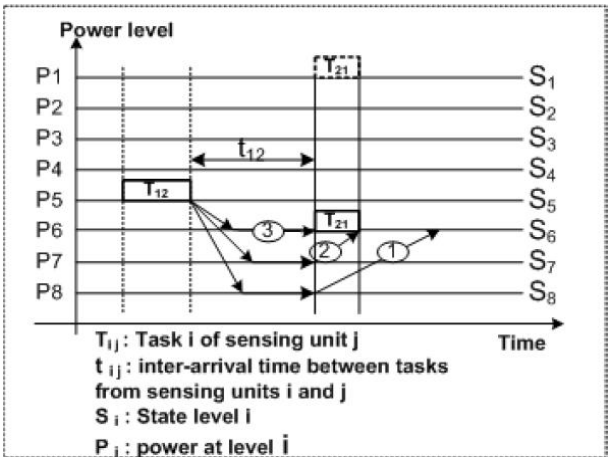

Fig. 7. The trade-off between energy savings achieved by selecting a low power state for a sensor node and event misses. If the sensor node switches to the state with lowest power consumption, it may miss the processing of the next event

The sleep mode period, T_s, is determined by the minimum duration an event of the corresponding type reaches its peak sensing signal strength value once the threshold for event detection is exceeded. The basic idea behind the proposed sensing operation of nodes is that even if an event occurs when the sensor node is temporarily in sleep mode, the sensor node switches back to active mode without missing the maximum signal strength of the event. The relationship between the characteristics of the observed phenomena and sleep modes of sensing activity is illustrated in Figure 4.

CTAS enables the neighboring nodes which keep their sensing units active for the same event type to utilize their region of overlap effectively for the coverage requirements. This is achieved by the arrangement of the times during which neighboring nodes switch to active mode for the common event type. For each event type assigned, the sensor nodes maintain the slots that a neighboring sensor node keeps its sensing unit active based on the contents of the SCMs received. The sensor node considers a slot as "occupied" with respect to an event type if any one of its neighboring nodes senses the environment for the corresponding event type in this slot. Then, if minimum coverage is required for this event type, the sensor node selects the starting slot index for its active mode of operation which results in the minimum overlap with the union of occupied slots. However, if the event type requires strong coverage of strength k, the sensor node makes its selection in such a way that its slots of active mode overlap with a number of occupied slots that is as close as possible to $(k-1)$ on the average. Figure 5 illustrates the slot selection approach used by node C with respect to its neighboring sensor nodes A and B that it shares an overlapping sensing region for the minimum coverage case. It is important to note that as T_s gets larger as compared to T_a, the efficiency of the slot selection approach will increase due to the larger number of candidate slots for starting active mode of operation. The proposed slot selection approach is based on the fact that the overlap of sensing regions is a drawback for event types requiring minimum coverage. However, for strong coverage requirements, the overlapping sensing regions may help in increasing the number of sensor nodes observing an event.

2) Data Transmission Task Scheduling: A sensor node keeps its sensing unit active and continues to sense events for an assigned event type. The sensor nodes cooperate with their neighboring nodes for the transmission of event data by advertising SIMs. After an event occurrence, the broadcast of the SIM message is scheduled according to the ratio of the residual energy level E_{res} of the node to its initial energy level E_0 in *Coordination Window*. However, in this case, the preference is given to high residual energy nodes to broadcast first. The cluster-head receives the SIM messages and selects a subset of nodes to transmit the data associated with the event based on reliability, timing requirements and residual energy of sensor nodes. The cluster-head can select the owners first K SIMs received based on the observation that those nodes have higher residual energy as compared to their neighboring nodes. In an alternative node selection policy, the cluster-head can request the sensor nodes with highest received sensing signal strength values to send data in order to achieve the most reliable measurements. The sensor nodes selected by the cluster-head transmit data while others suppress their transmission. In the rest of the section, we present the details of fine grain scheduling within sensor nodes.

Algorithm 2 *CTAS*

Require: ρ : The coverage QoS threshold in percentage
Ensure: Scheduling of sensing and data transmission tasks of a sensor node

for all round R of constant duration T in network lifetime **do**
  $M \leftarrow$ Ids of node's sensing units sorted in descending order according to the energy consumption of the corresponding sensing units in *FineGrainTaskScheduler* in the previous round
  $r_{s(i)} \leftarrow$ Sensing range for sensing unit i, $i = 1..|M|$. *// Incorporates any changes in sensing strength*
  $N_i \leftarrow$ List of active sensor nodes for each sensing unit that have overlapping sensing regions with the node, $i = 1..|M|$
  Initialize TDM to all 0s *// Initially Task Delegation Mask is empty and all sensing units are operational*
  Update active neighbor lists N_i for each SCM received in the rest of the round, $i = 1..|M|$
  for all i = 1 .. |M| **do**
    if *CoverageTest* (N_i, $r_{s(i)}$) $\geq$ ρ% **then**
      // Sensing units with their sensing regions completely covered
      Set TDM[i] to 1 *// Indicate that this sensing unit will no longer be active*
      Shut Down the i^{th} sensing unit
    else
      Activate i^{th} sensing unit
    end if
  end for
  Set broadcast timer to $\lfloor W * \frac{E_{res}}{E_0} \rfloor$
  Advertise SCM message upon timer expiration *// Notify neighboring nodes on sensing units that will be turned off*
  for all i = 1 .. |M| **do**
    // For each deactivated sensing unit, verify that active neighbors did not modify their decision
    if $CoverageTest(N_i, r_{s(i)}) < \rho\%$ **and** TDM[i] = 1 **then**
      Activate sensing unit corresponding to M_i
    end if
  end for
  for all event e observed **do**
    // Fine grain scheduling of event processing tasks within the sensor node
    Call *FineGrainTaskScheduler*(e, TDM) for the processing of event
    // Data transmission task scheduling for the event
    Set event type, sensing signal strength and priority fields in SIM
    Set broadcast timer to $\lfloor W * \left(1 - \frac{E_{res}}{E_0}\right) \rfloor$ *// Give priority to high residual energy nodes to process the event*
    Broadcast SIM message upon timer expiration
    if sensor node is selected for transmission by cluster-head **then**
      Transmit data for the event to the cluster-head
    else
      Cancel data transmission for the event
    end if
  end for
end for

E. Fine Grain Scheduling

The sensor nodes follow the *FineGrainTaskScheduler* for scheduling tasks at individual nodes. The fine-grain scheduler uses the priorities and deadlines of the tasks as well as the remaining execution time left for the current task to decide whether to replace the current task with a new task. If the current task will be replaced, the sensor node waits until any ongoing data transfer operation between Analog-to-Digital Converter, memory, CPU and RF unit completes before preempting the task. The sensor node maintains a buffer in which the preempted tasks are queued to be completed. The *FineGrainTaskScheduler* interacts with the coarse-grain scheduling phase of the algorithm by taking the TDM as an input parameter. The TDM has been advertised by the sensor node (within an SCM) at the beginning of the round for the coarse-grain scheduling and contains information on which sensing units of the node will be active. As we will explain, this information is used by the fine-grain scheduler to select the power mode of the sensor node. In return, the fine-grain scheduling phase provides information on the energy spent for each event type. The coarse-grain scheduling phase tries to offload the event types with the highest energy consumption to the neighboring nodes in the next round while preserving the coverage QoS for each event type.

FUNCTION 3 *FineGrainTaskScheduler*

Require: e : The event to be processed
TDM : The Task Delegation Mask advertised by the sensor node for the current round

Ensure: The scheduling of event processing tasks within the sensor node

1: $task_i \leftarrow$ The task currently in execution at the sensor node
2: A new task ($task_j$) is created for e
3: $D_j \leftarrow$ the deadline of $task_j$
4: $T_{a(j)} \leftarrow$ the time needed for processing of $task_j$
5: $t \leftarrow$ current time
6: **if** $task_j$ has higher priority than $task_i$
and $(t + T_{a(j)}) \simeq D_j$
and (the time overhead of switching from $task_i$ to $task_j$) $<$ (the remaining time left for $task_i$) **then**
7: The data transfer operations that are in progress for $task_i$ are completed
8: $task_j$ is assigned the resources relinquished by $task_i$
9: The sensor node starts processing $task_j$
10: $task_i$ is queued according to its priority
11: **else**
12: $task_j$ is queued according to its priority
13: The processing of $task_i$ is resumed
14: **end if**
15: The processing of queued tasks are completed during which the power mode of the sensor node is assigned to the state needed for the next task in the queue
16: The sensor node determines the event type of each completed task and updates the total energy consumption information for the event type
17: The sensor node selects the power state with respect to the event type that satisfies the following two properties
- The corresponding bit of the event type is set to zero in TDM
- The sensing unit for this event type has the minimum sleeping time (T_s) left since its last activation among other active sensing units

The task for the processing of an event detected not only contends for the limited resources of a sensor node with other events, possibly of different types, but also requires sensor node to be in a particular power state among the valid power states $\{S_1, S_2, ..., S_8\}$. Therefore, we first define the properties of the transitions between power modes of a sensor node before elaborating on fine grain task scheduling. Let us denote a high power state as S_h and low power state S_l as shown in Figure 6 where $1 \leq h < l \leq 8$ with respect to the states given in Table I.

- *The time overhead:* The sensor node spends the time, τ_{hl}, while transitioning from S_h to S_l. The time it takes to do the reverse transition is denoted by τ'_{lh} with the property that $\tau'_{lh} > \tau_{hl}$. The latency for a state transition increases with the increasing difference between the power consumption of the modes involved in the transition, that is, $\tau'_{lk} \geq \tau'_{lh},\ 1 \leq k \leq h < l \leq 8$.
- *The energy overhead:* The switching time between the initial and final power modes not only affects the event misses but also determines the energy overhead of the transition. The transition energy for switching to a less active state is modelled as $\delta_{hl} = \left(\frac{(P_h+P_l)}{2}\tau_{hl}\right)$. Similarly, $\gamma_{lh} = \left(\frac{(P_h+P_l)}{2}\tau'_{lh}\right)$ represents the energy overhead for transitioning to a more active power state. It can be observed that transitioning from a less active state to a more active state is more costly than its reverse transition ($\gamma_{lh} > \delta_{hl}$) based on the relationship of τ_{hl} and τ'_{lh}.

CTAS selects the power mode for a sensor node by considering the properties above in such a way that the sensor node benefits from the trade-off between the energy savings based on the usage of low power states and event misses as much as possible. This tradeoff is illustrated in Figure 7 which shows S_5, the power mode used by the sensor node for the processing of the current event, e_{curr}, and the three alternative low power states for the sensor node S_6, S_7, S_8 that it can switch to when the processing of e_{curr} is completed. It can be observed that if the next event, e_{next}, requires the sensor node to be in S_6, putting the sensor node in S_8 will cause the sensor node to miss the deadline for the processing of e_{next}. This is because of the larger latency involved in switching from S_8 to S_6. Similarly, if e_{next} requires sensor node to be in S_1, the sensor node may not be put in any one of these low power states due to the increased transition latency from these modes to S_1.

The sensor node selects the power mode according to the next task in the queue while executing preempted tasks. If there is not any remaining task in the queue, the fine grain scheduler exploits the sleep duration, T_s, associated with each sensing unit to select the best power mode for the sensor node. In particular, the sensor node determines the sensing unit with the minimum sleep duration left since its last activation. This

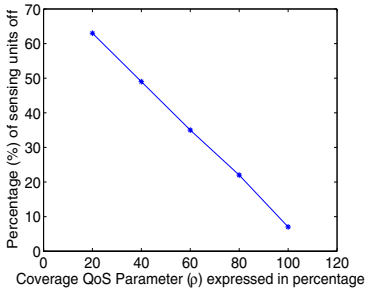

Fig. 8. The effect of coverage QoS requirement on the percentage of active sensing units

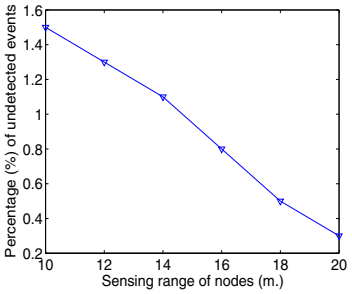

Fig. 9. The improved sensing capabilities of nodes helps CTAS to reduce undetected events

is based on the observation that the sensor node may have to process the next event from this sensing unit.

V. Performance Evaluation

This section first presents the performance analysis of CTAS algorithm followed by its experimental evaluation. The energy savings achieved in each round by the coarse-grain scheduling phase is analyzed first since communication is a key component in the overall energy consumption of nodes. We consider a group of sensor nodes covering the target region A, that are within the transmission range of each other. We assume that the cluster-head selects only one sensor node for data transmission after each event occurrence. The following terminology is used in the analysis.

- N : The total number of nodes in the group.
- $N_{act(i)}$: The total number of active sensing units for event type i with the completion of coarse-grain scheduling, $1 \leq i \leq |M|$.
- $N_{opt(i)}$: The minimum number of active sensing units that are needed to cover A for event type i, $1 \leq i \leq |M|$.
- λ_i : The average rate of event occurrences in A for event type i based on the assumption that temporal behavior of events is a Poisson process, $1 \leq i \leq |M|$.
- m_{SIM} : The SIM message size.
- m_{SCM} : The size of SCM messages.
- $m_{d(i)}$: The data packet size for event type i, $1 \leq i \leq |M|$.
- R : The data transmission rate of the sensor radio.
- P_r : The receive power of the radio.
- P_t : The transmission power of the radio.

The total energy consumption of the sensor nodes with coarse-grain scheduling is expressed as

$$E_{coarse} = \left(\frac{N m_{SCM}}{R} ((N-1)P_r + P_t) + E_{dtsch} \right)$$

where the first and second terms represent the energy consumption of event-type scheduling and data transmission scheduling for all event types respectively. After each event occurrence, the sensor nodes first transmit SIM messages to the cluster-head. Then, the cluster-head indicates the sensor node selected for data transmission and finally the data is transferred to the cluster-head. Thus, E_{dtsch} can be further expanded into

$$\sum_{i=1}^{|M|} \lambda_i T \left[\begin{array}{l} P_t \left(\frac{\lceil \frac{N_{act(i)}}{N_{opt(i)}} \rceil m_{SIM} + \lceil \log_2 N \rceil + m_{d(i)}}{R} \right) \\ P_r \left(\frac{\lceil \frac{N_{act(i)}}{N_{opt(i)}} \rceil (m_{SIM} + \lceil \log_2 N \rceil) + m_{d(i)}}{R} \right) \end{array} \right]$$

The total energy consumption without the application of coarse-grain scheduling would be

$$\sum_{i=1}^{|M|} \left[\lambda_i T (P_r + P_t) \frac{\left(\lceil \frac{N}{N_{opt(i)}} \rceil m_{d(i)} \right)}{R} \right].$$

The key factor for the energy savings achieved is the difference between N and $N_{act(i)}$, $1 \leq i \leq |M|$, in addition to the use of short control packets for data aggregation ($m_{d(i)}$-m_{SIM}). Thus, scheduling of tasks based on the coverage based relationship of sensor nodes is an effective approach for improving the energy consumption of the network.

We next evaluate the delay characteristics by considering the data transmission durations only and ignoring the propagation delays. In the absence of coarse-grain scheduling, the delay for the cluster-head to receive the data for an event of type i, $1 \leq i \leq |M|$, is

$$\sum_{i=1}^{n} \frac{m_{d(i)}}{R}$$

where n is equal to $\lceil \frac{N}{N_{opt(i)}} \rceil$. The use of small sized SIM messages improves the data transmission delay of CTAS which is expressed as

$$\frac{(\sum_{i=1}^{n} m_{SIM}) + \lceil \log_2 N \rceil + m_{d(i)}}{R}$$

The experimental evaluation of CTAS is performed by extending PowerTOSSIM [20]. PowerTOSSIM is a scalable simulation environment for wireless sensor networks which provides an accurate, per-node estimate of the energy consumption with a detailed model of MICA-2 mote hardware [18]. In the simulations, the temporal behavior of events over the target area is modelled with a Poisson process and the locations of events are randomly selected. We have evaluated the performance of CTAS in terms of energy efficiency, event misses, quality of coverage service and realtime data delivery

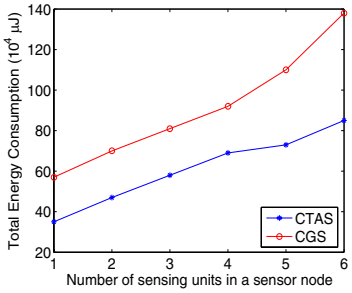

Fig. 10. CTAS resolves the contention for limited resources of a sensor node with fine grain scheduling of tasks

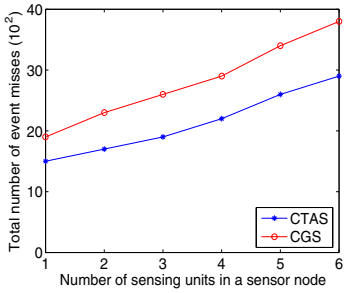

Fig. 11. CTAS reduces event misses by selecting the power modes of sensor nodes according to the event processing requirements

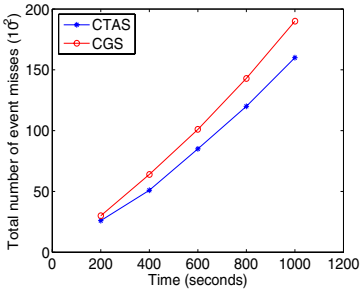

Fig. 12. The performance of CTAS in terms of event misses as network gradually consumes its energy

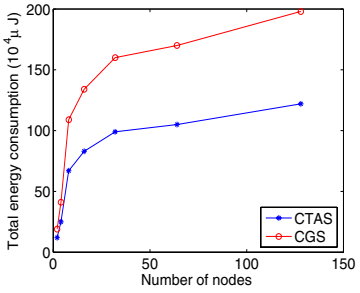

Fig. 13. The total energy consumption due to the processing and transmission of redundant data is mitigated with the application of CTAS

performance. CTAS is the first of its kind in its cross level scheduling of tasks by exploiting complete / partial overlaps among the sensing regions of the nodes. Therefore, we developed an algorithm named CGS (*C*overage Redundancy Based *G*reedy Task *S*cheduling) for comparison purposes. CGS makes use of the coverage scheduling technique proposed in [4] for coarse-grain scheduling. In fine-grain scheduling phase, CGS selects the lowest power state following a greedy approach whenever there are no tasks to be executed and does not differentiate an event type from others while scheduling tasks. The default data packet size in PowerTOSSIM, 29 bytes, is used among with 2 bytes long control packets. In the simulation results, we refer an event as *undetected* if its location is outside the sensing regions of all active sensing units at the time of event occurrence. We note the difference from the missed events that are defined as the events detected but could not be processed due to the power modes selected by the sensor nodes.

We first evaluate the performance of CTAS with respect to the key system parameters. The algorithm enables users to trade off coverage QoS for the target area with the energy consumption of the sensor group using parameter ρ. Figure 8 shows the effect of coverage QoS on the percentage of sensing units that can be turned off. It can be observed that strict coverage requirements force more sensor nodes to stay active in order to reduce the coverage breaches in the target region. Figure 9 depicts the results of the simulations performed with constant node group size of 128 and varying the sensing range of the nodes. As can be observed from the figure, CTAS exploits the enhanced capabilities of sensor nodes to reduce the number of undetected events while reducing coverage redundancy.

In the first set of experiments for the comparison of CTAS with CGS, the number of sensing units in sensor nodes is increased which results in high contention for the limited node resources. The addition of new sensing units also increases event misses since the probability that a sensor node is busy with the processing a task when a new event occurs becomes higher. Figure 10 and Figure 11 illustrate the fact that CTAS significantly improves the energy consumption of the nodes and mitigates the event misses. CTAS achieves these savings with the selection of best power modes for the sensor nodes, scheduling of tasks using event priorities and exploiting characteristics of event occurrences. The sensor network gradually consumes energy, thus, the performance of the algorithms in the long run should also be evaluated. To this end, Figure 12 shows the change in the total number of event misses with respect to time.

The key feature of CTAS is its ability to control the coverage redundancy while preserving the quality of coverage service over the target area. Figure 13 shows the average results for simulation runs in which the group size is increased for the same target area. As the node population is increased, more sensor nodes observe the same event in CGS. In comparison, CTAS keeps the sensing units of a smaller subset of nodes active, therefore, the processing and data transmission overhead for redundant data is reduced with its application. Figure 14 illustrates how the energy consumption for a group of 128

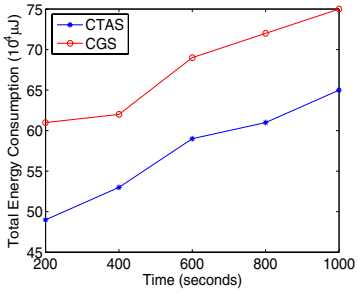

Fig. 14. The total energy consumption of the CTAS algorithm with respect to time

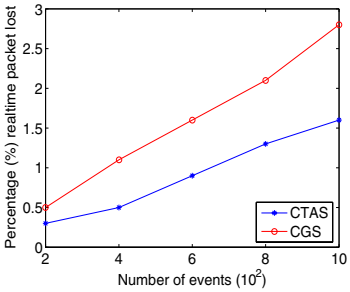

Fig. 15. CTAS meets the deadline requirements of realtime data packets with the use of short control packets

nodes changes with respect to time with the application of CTAS and CGS.

The algorithm is also evaluated in terms its realtime data delivery success rate, which is defined by the percentage of data packets that are delivered to the cluster-heads before their deadline. In the last set of experiments, the number of events is increased for a group of sensor nodes equipped with 6 sensing units for the same simulation time. As can be observed from Figure 15, the use control packets enables CTAS to achieve a high realtime data delivery success rate as compared to CGS.

VI. Conclusions

We have presented a two-level task scheduling algorithm, CTAS, for a group of sensor nodes with multiple sensing units. CTAS first schedules event types for sensor nodes and, then, determines which nodes should transmit data to the cluster-head. The fine-grain scheduling part of CTAS schedules tasks within each sensor node for the assigned event types. A new coverage scheme is described to determine the degree of overlap between sensing regions of neighboring nodes that is utilized in the coarse-grain scheduling phase. We also addressed the problem of selecting the best power mode for sensor nodes in fine-grain scheduling phase. The simulation results show the effectiveness of the proposed algorithm in terms of reduction in energy consumption and event misses in the sense that energy consumption is reduced up to 67% and the event misses are decreased by 75%.

References

[1] B. Chen, K. Jamieson, H. Balakrishnan and R. Morris, "Span: An Energy-efficient Coordination Algorithm for Topology Maintenance in Ad Hoc Wireless Networks", *Proc. of the Seventh Annual International Conference on Mobile Computing and Networking*, pp. 85-96, July 2001.

[2] Y. Xu, J. Heidemann and D. Estrin, "Geography-informed Energy Conservation for Ad Hoc Routing", *Proc. of the Seventh Annual International Conference on Mobile Computing and Networking*, pp. 70-84, July 2001.

[3] Ye, W., J. Heidemann and D. Estrin, "An Energy-efficient MAC Protocol for Wireless Sensor Networks", *Proc. of the 21st International Annual Joint Conference of the IEEE Computer and Communications Societies*, Vol.3, pp. 1567-1576, June 2002.

[4] Di Tian and Nicolas D. Georganas, "A Coverage-Preserving Node Scheduling Scheme for Large Wireless Sensor Networks", *Proc. of the 1st ACM International Workshop on Wireless Sensor Networks and Applications*, September 2002.

[5] H. O. Sanli and H. Çam, "Energy Efficient Differentiable Coverage Service Protocols for Wireless Sensor Networks", *Proc. of Third IEEE International Conference on Pervasive Computing and Communications Workshops*, pp. 406-410, March 2005.

[6] H. O. Sanli, H. Çam and X. Cheng, "EQoS: An Energy Efficient QoS Protocol for Wireless Sensor Networks", *Proc. of Western Simulation MultiConference*, January 2004.

[7] T.Yan, T. He and J. A. Stankovic, "Differentiated Surveillance Service for Sensor Networks", *First ACM Conference on Embedded Networked Sensor Systems*, 2003.

[8] C. Lu, B. Blum, T. F. Abdelzaher, J. Stankovic and T. He, "RAP: A Real-Time Communication Architecture for Large-Scale Wireless Sensor Networks", *Proc. of Eight IEEE Real-Time and Embedded Technology and Applications Symposium*, 2002.

[9] A. Mahapatra, K. Anand and D. Agrawal, "QoS and energy aware routing for real-time traffic in wireless sensor networks", *Special Issue of the Journal of Computer Communications on Sensor Networks*, 2004.

[10] R. Cristescu and M. Vetterli, "On the Optimal Density for Real-Time Data Gathering of Spatio-Temporal Processes in Sensor Networks", *Proc. of the Fourth International Conference on Information Processing in Sensor Networks*, 2005.

[11] Y. Yu, B. Krishnamachari and V. K. Prasanna, "Energy-Latency Tradeoffs for Data Gathering in Wireless Sensor Networks", *Proc. of Twenty-third Annual Joint Conference of the IEEE Computer and Communications Societies*, Vol. 1, pp. 244-255, 2004.

[12] K. Akkaya and M. Younis, "An Energy-Aware QoS Routing Protocol for Wireless Sensor Networks", *Proc. of Twenty-third International Conference on Distributed Computing Systems Workshops*, 2003.

[13] R.Poornachandran, H.Ahmad and H. Çam, "Energy-Efficient Task Scheduling for Wireless Sensor Nodes with Multiple Sensing Units", *Proc. of IWSEEASN'05*, April 2005.

[14] A. Sinha and A.Chandrakasan, "Operating System And Algorithmic Techniques for Energy Scalable Wireless Sensor Networks," *Proc. of Mobile Data Management, Second International Conference*, January 2001.

[15] G. A. McIntyre and K. J. Hintz, "Sensor Measurement Scheduling: An Enhanced Dynamic, Preemptive Algorithm", *Optical Engineering*, Vol.37, no.2, pp.517-23, February 1998.

[16] A. R. Shahani, D. K. Schaeffer and T. H. Lee, "A 12 mW Wide Dynamic Range CMOS Front-End for a portable GPS Receiver", *Digest of Technical Papers, IEEE International Solid State Circuits Conference*, Vol. 40, pp. 368-369, February 1997.

[17] C. Savarese, K. Langendoen and J. Rabaey, "Robust Positioning Algorithms for Distributed Ad-hoc Wireless Sensor Networks", *Proc. of the USENIX Annual Technical Conference*, pp. 317-27, June 2002.

[18] http://www.xbow.com

[19] P. H. Chou, J. Liu, D. Li and N. Bagherzadeh, "IMPACCT: Methodology and Tools for Power-Aware Embedded Systems", *Design Automation for Embedded Systems, Special Issue on Design Methodologies and Tools for Real-Time Embedded Systems*, 7(3), pp. 205-232, October 2002.

[20] http://www.eecs.harvard.edu/shnayder/ptossim/