International Journal of Research in Advent Technology (IJRAT) Special Issue
E-ISSN: 2321-9637
Available online at www.ijrat.org
*International Conference "*INTELINC 18*", 12th & 13th October 2018*

Performance of Traffic MAC-Protocol Based on TCP Congestion avoidance algorithm for wireless Network

K.Vijayalakshimi, Dr.R.Manikandan Ph.d.
Computer Science and Engineering, Assistant Professor in CSE, Srinivasa Subbaraya Government Polytechnic college,Puthur, Department of Assistant Professor, Computer Science and Engineering,Government college of Engineering, Dharmapuri,
mailtovijius@gmail.com, rmkmanikandan@yahoo.co.uk.

Abstract :EA-MAC protocol is proposed on the basis of SMAC protocol to remedy the shortcomings of SMAC. In the EA-MAC the active node selection problem aims at selecting a minimum number of nodes to provide required data services within error threshold so as to efficiently extend the network lifetime. In this paper, we firstly propose a new Cover Sets Balance (CSB) algorithm to choose a set of active nodes with the partially ordered tuple (data coverage range, residual energy). Then, we introduce a new Correlated Node Set Computing (CNSC) algorithm to find the correlated node set for a given node. Finally, we propose a TCP's congestion avoidance algorithm and adjusts the transmission rate when a collision occurs, while the second one shifts packet transmission times to minimize collisions. We evaluated both algorithms through simulations and our results show that our approach can reduce the number of collision-induced retransmissions by a factor of 8 and the energy consumption by up to 50%.

1. INTRODUCTION

Current research in wireless sensor networks (WSN) mainly focus on data fusion, routing protocol, topology control, and medium access control (MAC) layer protocols [1, 2]. The main objective is to reduce the energy consumption of the sensor network, increase throughput, and reduce the data transmission delay. Today such networks are used in many industrial and consumer applications, such as traffic data collection, vehicular monitoring and control, security surveillance, and smart homes. Each sensor node is equipped with a sensing device which can detect the environmental condition. The nodes are also powered by limited batteries and it is difficult or impossible to replace them in some special environments. It is why energy efficiency is always the most important criterion for such networks. One important approach to extend the network lifetime is to reduce the number of required packet transmissions in the network [2-5], such as clustering [6–11], in-network data aggregation [12–18], and approximate data collection [19, 20]. In these scenarios, all nodes in the network are considered active and the data are gathered from all nodes during the collecting process.

However, it is not an efficient way to collect all raw data from each node in some special applications which aim to collect information originated from the environment, such as temperature, humidity, and pressure. In these applications, it is fully tolerant if the final collected information is just within error threshold. The sensing data of each node is generally a noise version of the observed phenomenon and there is a deviation among them due to distance, location, or node sensitivity. Nodes are generally correlated if they are observing the same physical phenomena. Correlations between nodes are described in some simple ways such as the maximum or minimum value between nodes [21]. In this paper, correlation occurs if the sensing data of simple node can be obtained from the other nodes. Accordingly, a subset of active nodes can be selected to provide the required sensing service within error threshold, and the rest nodes can go to sleep and preserve energy. In this way, the active node selection strategy with correlation optimization not only prolongs the network lifetime, but also helps to solve other issues in dense wireless sensor networks [22], such as lower network throughput, serious node conflict, and excessive packet transmissions. How to describe the correlation among the sensing data quantitatively is the key issue when achieving an efficient active node selection strategy. The distance function is generally considered as an important model to formulate the data similarity between nodes because the sensitivity is sometime related to the distance between the source and sensing device. Here we adopt Manhattan distance between sensing data as error metric [22]. Based on the observation that the sensing data are similar to each other if they are close enough, Kotidis [23] proposes Snapshot query in which only selected active nodes report their sensing data, and sensing data of one-hop nodes is computed by active nodes. Liu et al. [24] propose an EEDC algorithm which divides the nodes into disjoint cliques based on spatial correlation so that the nodes in the same clique have similar sensing data and can communicate directly with each other. Hung et al. [22] propose a DC global algorithm to determine a set of active nodes with high energy levels and wide data coverage ranges. Figures 1(a) and 1(b) show the selected nodes with EEDC and DC global for a given wireless sensor network, where each circle denotes one node (the sensing data value is marked above the circle). The edge

between a pair of nodes denotes that they can communicate directly with each other. Here we assume that Manhattan distance is used as the similarity function and the error threshold is 0.5. The selected active nodes are marked with black solid circle. The selected active node set with EEDC is {s1,s2,s5,s6,s8,s9,s12}{s1,s2,s5,s6,s8,s9,s12},and it is {s1,s5,s10,s13}{s1,s5,s10,s13} with DC global.According to Figure 1, the number of selected nodes with DC global is smaller than EEDC in this example

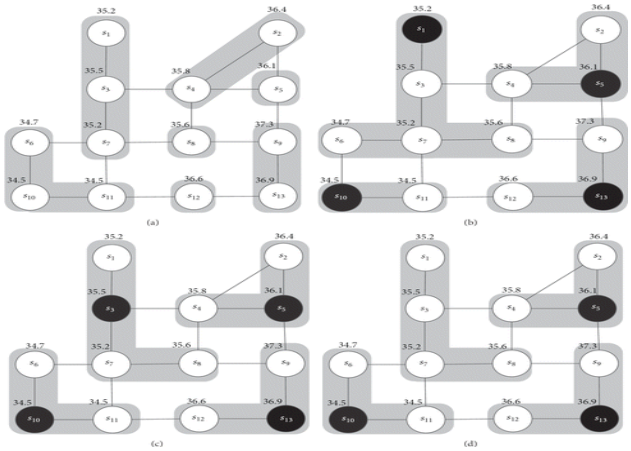


Figure 1 An example to demonstrate different algorithms. (a) EEDC, (b) DCglobal, (c) CSB, and (d) TCP.

The concept of data coverage range is firstly introduced to describe the correlation among nodes and defined as a node set in which the distance between each element and the given node is within the error threshold [22]. In fact, it is a simple extension of one-hop data coverage [23]. Another issue of [22] is the efficiency of proposed node selection algorithm. The partially ordered tuple (residual energy, data coverage range) is used to select an active node set, which ensures that the selected nodes always have high reserved energy, but the number of selected active nodes is not minimized.

To address these problems, we introduce several new concepts, that is, cover set, active node, and covered node, and propose a new Cover Sets Balance algorithm (CSB) to choose a set of active nodes with wide data coverage range and high energy level by using the partially ordered tuple (data coverage range, residual energy) and build the corresponding cover set in sequence to ensure the selected active nodes have high residual energy. In this way, the set of final selection nodes generally owns larger residual energy and smaller size, which helps to extend the network lifetime. Figure 1(b) demonstrates the set {s1,s5,s10,s13}{s1,s5,s10,s13} generated by DC global assuming that reserved energy is identical to all nodes in the network, which is similar to the partially ordered tuple (data coverage range, residual energy). Figure 1(c) demonstrates the result as {s3,s5,s10,s13}{s3,s5,s10,s13} with the proposed CSB algorithm in this paper. {s1,s7,s8}{s1,s7,s8} is a cover set for node s3s3 and each node in the set is a feasible candidate regarding s3s3.

In the following we show some nodes can be further removed from the selected active node set with CSB. As shown in Figure 1(c), the sensing data of s3s3, s5s5, s10s10 is 35.5, 36.1, and 34.5, respectively, the average value of s5s5 and s10s10 is 35.3. The Manhattan distances between 35.3 and sensing data s1s1, s3s3, s7s7, and s8s8 are 0.1, 0.2, 0.1, and 0.3 accordingly (they are all less than the error threshold 0.5). It means that the sensing data of CS3+{s3}CS3+{s3} is computed by s5s5 and s10s10. Accordingly, s3s3 is removed and then we have a smaller active node set {s5,s10,s13}{s5,s10,s13}, as shown in Figure 1(d). Following this observation, we introduce a novel concept Correlated Node Set (CNS) and then we propose a TCP's congestion avoidance algorithm(TCP) node selection algorithm to reduce the number of active nodes. The main contributions of this paper are as follows.

(i)We propose a Cover Sets Balance algorithm (CSB) to select a set of active nodes with wide data coverage ranges and high energy levels. In each active node selection step, we use the partially ordered tuple (data coverage range, residual energy) to find an initial active node set and then balance the size of the cover sets in order to replace low-energy nodes.

(ii)We propose a Correlated Node Set Computing algorithm (CNSC) to calculate the correlated node set with minimum set size and maximum geometric mean of residual energy of each node in the sensor network by following the observation that some nodes selected by CSB can be further removed.

(iii)We propose a TCP's congestion avoidance algorithm (TCP) to reduce the number of active nodes selected with CSB by removing nodes which can be computed by correlated node sets.

The rest of this paper is organized as follows. In Section 2, we describe the system model. Section 3 introduces CSB and TCP algorithms. The theoretical analysis of the algorithms is proposed in Section 4. In Section 5, we describe the simulation results and performance analysis. Section 6 presents the related works and Section 7is conclusion.

2. SYSTEM MODEL

A wireless sensor network generally consists of a set of stationary nodes V={s1,s2,…,sn}V={s1,s2,…,sn}, and each node in the network has identical transmission radius r. The network is formulated as an undirected graph G=(V,E)G=(V,E) with V as the set of nodes and E as the set of links. Without loss of generality, both sisi and i are used to represent one single node in the network. There is a link (i,j)(i,j) between node i and j if they communicate with each other directly.

The nodes are equipped with unreplaceable or unrechargeable batteries. The reserved energy for node i at time t is denoted by ei(t)ei(t).

The collected data from one single node is a noise version of the practical phenomena. In these applications, the collected information from the sensor network is tolerant in case that it is within a given error threshold ε.

The notations used in this work are listed as the following:

V:V: Set of nodes in the network

n: Number of nodes in the network

ε: One given error threshold

xi(t)xi(t): Sensing data of i at time t

d(xi(t),xj(t))d(xi(t),xj(t)): Distance between the sensing data of i and j

Energyi(t)Energyi(t): Residual energy of i at time t

DCRiDCRi: Data coverage range of i

CSiCSi: Cover set of i

CNSiCNSi: Correlated node set of i

ANSANS: Active node set

r: Transmission radius

maxdmaxd: Maximal number of nodes in a correlated node set

Eventj(t) Eventj(t): Value of event j at time t

Interval: Interval to reselect a new active node set.

The correlation among sensing data especially in a dense wireless sensor network is helpful to extend the network lifetime. Some researchers studied the correlation between nodes and provided some models [25]. Among all these models, it is common to adopt distance function, such as Manhattan distance d(xi(t),xj(t))d(xi(t),xj(t)) to represent the correlation between sensing data xi(t)xi(t) and xj(t)xj(t) at time t [22], which is represented as d(xi(t),xj(t))=||xi(t)−xj(t)||d(xi(t),xj(t))=|xi(t)-xj(t)|.

Without loss of generality, we follow this correlation model in this paper. Note that our algorithms are adapted to any other correlation models with minor modification.

The sensing data of sjsj is called to be computed with the sensing data of sisi if the sensing data has a high correlation level; that is, d(xi(t),xj(t))≤εd(xi(t),xj(t))≤ε, where ε is the given error threshold. SjSj is also in the data coverage range of sisi. The definitions are given as follows.

Definition 1 (data coverage range (DCR)).
Given an error threshold ε in the sensor network, the data coverage range DCRiDCRi of iis a subset of V, in which Manhattan distance from each node to i is no more than ε, and i∉DCRii∉DCRi.

For the example in Figure 1(b), ε=0.5ε=0.5, DCR1={s3,s6,s7,s8}DCR1={s3,s6,s7,s8},

DCR3={s1,s4,s7,s8}DCR3={s1,s4,s7,s8}, and so on.

Definition 2 (active node set (ANS) and active node).
Given a sensor network G=(V,E)G=(V,E), an active node set (ANS) is a subset of V, in which each i∈Vi∈V either belongs to ANS or one data coverage range DCRjDCRj, where j∈ANSj∈ANS. Any node in ANS is named as an active node.

For the example in Figure 1(c), ANS={s3,s5,s10,s13}ANS={s3,s5,s10,s13} and each node in the set is an active node.

***Definition 3* (cover set (CS) and covered node).**
Given a sensor network G=(V,E)G=(V,E) and according ANS, the cover set CSiCSi for any given i∈ANSi∈ANS is a subset of DCRiDCRi, and CSi∩CSj=∅CSi∩CSj=∅ in case i≠ji≠j. Any node in CSiCSiis named as a covered node.

For the example in Figure 1(c), CS3={s1,s7,s8}CS3={s1,s7,s8}, s3s3 is the active node, and s1s1, s7s7, and s8s8 are covered nodes.

Sensor data is affected by the events in monitored region, and the influence of each event on a sensor is inversely proportional to their distance. Here we assume that correlation occurs among all active nodes in the sensor network.

Definition 4 (correlated node set (CNSCNS)).
Given a sensor network G=(V,E)G=(V,E) and its corresponding ANSANS, the correlated node set CNSiCNSi for i∈ANSi∈ANS is a subset of ANSANS, and the arithmetic mean s−s- for sensing data of nodes in CNSiCNSi satisfies the error threshold condition; that is, d(xj(t),s−)≤εd(xj(t),s-)≤ε, j∈CSi+{i}j∈CSi+{i}.

The sensing data of CSi+{i}CSi+{i} is said to be computed by CNSiCNSi.

For the example in Figure 1(d), CNS3={s5,s10}CNS3={s5,s10}, s−=35.3s-=35.3, d(x1,s−)=0.2≤εd(x1,s-)=0.2≤ε, d(x7,s−)=0.1≤εd(x7,s-)=0.1≤ε, d(x8,s−)=0.3≤εd(x8,s-)=0.3≤ε, d(x3,s−)=0.2≤εd(x3,s-)=0.2≤ε.

Definition 5 (CNS computing problem).
Given a sensor network G=(V,E)G=(V,E), ANS, sensing

data X={x1,x2,…,xn)X={x1,x2,…,xn), cover sets CS=(CS1,CS2,…,CSn)CS=(CS1,CS2,…,CSn), and reserved energy Energy={e1,e2,…,en}Energy={e1,e2,…,en}, the CNS computing problem is to find a correlated node set CNSiCNSi for node i∈ANSi∈ANS, and the size of |CNSi||CNSi| is minimized while the geometric mean of residual energy eˆe^ is maximized, where eˆ=e1e2⋯en————————√ne^=e1e2⋯enn.

Note that we adopt the geometric average of the residual energy in the correlated node set by following the observation that the average geometric averaging gives higher results for lower variations in the data values for a given data set with a fixed arithmetic.

Definition 6 (active node selection problem).
Given a sensor network G=(V,E)G=(V,E), the sensing data X={x1(t),x2(t),…,xn(t)}X={x1(t),x2(t),…,xn(t)}, reserved energy levels Energy(t)={e1(t),e2(t),…,en(t)}Energy(t)={e1(t),e2(t),…, en(t)} at time t and given threshold ε, the active node selection problem is to select a set of active nodes ANS(t)ANS(t) at time t, where all sensing data in the network can be computed by their corresponding active nodes, and the network lifetime is maximized, that is, max{t}max{t}.

The active node selection problem is to find the active node set during each epoch and aim at maximizing the network lifetime. The problem is proven to be NP-hard by mapping it to the set covering problem or minimum dominating set problem .In this paper, we design two heuristic algorithms, namely, CSB and TCP for this problem.

Most related works use the concept of data coverage range combined with energy to solve the active node selection problem. In this section we illustrate the Cover Sets Balance algorithm (CSB) based on the idea of data coverage range.In data collection process, only active nodes are required to provide perception service, and the rest nodes are closed to preserve energy. An intuitive approach for the node selection process is to use the partially ordered tuple (data coverage range, residual energy) [23]. Another approach is to use partially ordered tuple (residual energy, data coverage range) to select active nodes with higher residual energy [22]. However, the number of selected nodes is generally larger than the former approach, which means that more energy consumption is necessary when providing perception service during the given epoch. Obviously, we need a balance between the two metrics, that is, the data coverage range and residual energy.

The basic idea of Cover Sets Balance (CSB) algorithm is described as the following: (1) generate an initial active node set and the corresponding cover sets through the previous data coverage range priority strategy; (2) replace active nodes with high-energy candidates. Note that the candidates must cover all nodes within the same cover set.

Figure1(b), CS1={s3,s6,s7,s8}CS1={s3,s6,s7,s8}, CS5={s2,s4}CS5={s2,s4}, CS10={s11}CS10={s11}, and CS13={s9,s12}CS13={s9,s12}. It is seen that s1s1 covers four nodes, namely, s3, s6, s7s3, s6, s7, and s8s8, and thus the candidate node for s1s1 must cover the above four nodes too. However, if the cover set is too large, it is possible to find no candidate nodes. The case is similar when the cover set is too small. Obviously we need a new method to provide more candidate nodes so that the network lifetime is extended in an efficient way.

We adopt a cover set balance strategy to balance the set size by moving nodes from larger cover sets to smaller ones. The initial cover sets are sequenced in descending order of the set size, and then we check nodes in one cover set and try to move them to another with smaller size. This process continues until all sets are checked and finally they are balanced. This strategy is helpful to increase the number of candidate nodes with higher residual energy by cutting down the maximal deviation of each cover set in the balance progress.

The final step of the CSB algorithm is to replace the selected active nodes with candidates by order of reserved energy. In this way, we finally build an active node set with the same size as its initial version but higher residual energy, which is helpful to extend the network lifetime.

The CSB algorithm can be divided into three processes and pseudocodes are shown in Algorithm 1.

Algorithm 1: Pseudocodes for Cover Set Balance (CSB) algorithm.
•Input: G = (V, E), ε, X = {x1,x2,…,xn}{x1,x2,…,xn}, Energy = {e1,e2,…,en}{e1,e2,…,en};
•Output: ANS, CS.
•(1) //Initialization_Process ( )
•(2) Calculate DCR = {DCR1, DCR2, …, DCRn}DCRn}
•(3) Set the state of all nodes as Un-Covered;
•(4) Sort nodes into sequence with partially ordered tuple ⟨data coverage range, residual energy⟩ ⟨data coverage range, residual energy⟩ ;
•(5) ANS ←∅, CS ←∅;
•(6) for one maximal i in the sequnce with state as Un-Covered
•(7) ANS ←{i}{i} + ANS, and set i as Primary-Covered;
•(8) for each j∈DCRiDCRi − ANS with state as Un-Covered
•(9) CSi←{j}+CSiCSi←{j}+CSi, and set j as Primary-Covered;
•(10) end for
•(11) end for
•(12) //Cover_Set_Balance_Process ( )

•(13) Sort CS into a sequence with decreasing order of the set size;
•(14) for each CSiCSi in the sequence
•(15)Sort nodes in CSiCSi with decreasing order of their deviation to i;
•(16) for each j in the sequence
•(17)find out all k which satisfies j∈CSkj∈CSk and |CSk|<|CSi||CSk|<|CSi|, select k with minimal cover set size;
•(18)CSi←CSi−{j}CSi←CSi-{j}, CSk←CSk+{j}CSk←CSk+{j};
•(19) end for
•(20) end for
•(21) //Node_Replace_Process ( )
•(22) for each i∈ANSi∈ANS
•(23) for each j∈CSij∈CSi
•(24) if d(xj,xk)≤εd(xj,xk)≤ε for any k∈CSi+{i}−{j}k∈CSi+{i}-{j}, then mark j as a candidate of i;
•(25) end for
•(26) select a node m from all candidates of i with maximal residual energy eiei;
•(27) ANS ← ANS + {m}−{i}{m}-{i}; CSm=CSi+{i}−{m}CSm=CSi+{i}-{m};
•(28) end for

The Initialization_Process is used to build a primary active node set and corresponding cover sets. The basic steps are described as follows. There are two different states for each node in the network, namely, Primary-Covered and Un-Covered, which are used to mark whether it is within the cover set of one node in the active node set. The states for all nodes are initialized as Un-Covered (Line 3). Then we sort nodes with partially ordered tuple (data coverage range, residual energy) and initialize the active node set as empty set (Line 4-5). Finally, we check nodes in sequence with state as Un-Covered, and add them into the active node set if the required conditions are satisfied (Line 6–11).

The Cover_Set_Balance_Process aims at balancing the size of cover sets generated with the Initialization_Process. Firstly, the cover sets are ordered and checked accordingly to their set size (Line 13). Secondarily, nodes in a given cover set CSiCSi are sorted into a sequence with descending order of their deviation to i (Line 15), and they are moved to another cover set with smaller size (Line 16–19). This process continues until all nodes in the cover set are checked (Line 14–20).

The Node_Replace_Process focuses on nodes exchange by replacing the low-energy active nodes with high-residual-energy candidates. All feasible candidate nodes of I are checked (Line 23–25), and we select the one (marked as m) with maximal residual energy among all these candidates (Line 26). Finally, the active node set is updated as well as the cover set for node m (Line 27).

The CSB algorithm follows the idea of replacing the active nodes with candidates with higher residual energy. However, it has the same number of active nodes compared with the approach which only uses the partially ordered tuple (data coverage range, residual energy). In the following we introduce a TCP-like approach, the receiver informs the source of a collision-induced loss. The source then reacts by multiplicatively decreasing its transmission rate. If no collisions are detected, then the source can increase its rate, thereby achieving lower latency over a static approach. In the phase offset approach, the source transmits at a constant rate. The receiver records packet arrivals and attempts to calculate the largest silent period between two consecutive packet transmissions. When a collision occurs, the receiver passes that information to the source which then changes its transmission phase to match the largest silent period. We compare and evaluate our algorithms using simulations. We show that, compared to a base case, in which sources transmit as fast as possible, our methods can decrease the number of collision-induced lost packets a factor of 8 and reduce additional energy consumption on the source by 50%.

Collision Avoidance

A simple and straightforward approach to reduce collisions is to use a probabilistic scheme. Time is partitioned into large epochs, in which each source randomly chooses a time to send. Since the epoch is sufficiently large compared to the packet send time, the probabil- ity of any two sources choosing overlapping send times is small. MOAP currently uses such a scheme.

By definition, each source can send only one packet per epoch. Therefore, latency is directly proportional to the size of the epoch. A sufficiently large epoch for a small network is much different than for a large net- work. In essence this scheme is not adaptive to the net- work environment. An adaptive solution should try to decrease latency when possible but also react to collisions as they occur. Ideally, it should lower the probability of future col- lisions without significantly increasing latency or the number of retransmissions.

Adaptive collision schemes can be categorized into two types:

source-based and receiver-based. In the source-based approach, there is no explicit feedback sent to the source from the receiver, except for that which is defined as a part of the protocol. An example of a source-based approach is Van Jacobson's TCP congestion avoidance [7]. TCP congestion avoidance uses the TCP protocol's duplicate ACKS and timeouts to infer network congestion and react to it.

The receiver-based approach allows the receiver to send feedback to the sender for the explicit purpose of reacting to collisions. As the receiver discovers pertinent information for collision avoidance, it transmits that information to the source. XCP [8], in which the network explicitly sends congestion information back to the source, is an example of a receiver-based approach.

Source-Based Collision Avoidance

Congestion avoidance on a packet switching network is similar to collision avoidance in a wireless network. In both cases the source throttles its data rate in the presence of an anomaly (congestion or collision). In collision avoidance, throttling the data rate changes the ar-rival period of the data packets, so any reoccurring collisions due to overlapping periods will be avoided. Also, in slowing down the source, heavy background traffic at the receiver is allowed to dissipate. This latter re- sult implicitly assumes that all source traffic has a fi- nite length and that after some time the background sources will eventually stop.

In the case of TCP congestion control, the design implicitly assumes that any packet loss is due to congestion. This assumption does not hold in wireless systems; data can be lost at the receiver from either collisions or link loss. Link loss encompasses a variety of reasons for packet corruption such as reflections, environmental changes, radio orientation, fading etc. Our assertion is that the source should not react to link losses since they are non-deterministic and outside the source's control. On the other hand collisions can be avoided if the source takes specific action. Since our source-based approach does not distinguish between link losses and collision losses, we expect this partic- ular scheme to be very sensitive to link quality. We call this method uninformed TCP-like collision avoidance.

Receiver-Based Collision Avoidance

As discussed above, the MOAP protocol al- ready sends NACKs to the source to inform it about lost packets. If the receiver can distinguish be- tween collision-induced losses and link losses then the source can use this information to make a better decision on whether to decrease the data rate or not. We call this method informed TCP-like collision avoidance and note that its performance de- pends on the accuracy of the collision detection scheme.

If we assume that all sources transmit packets at the same rate 1/T —that is, during the time frame T , every active source sends a packet—then the receiver can also monitor packet arrivals from all sources and detect large periods of inactivity within a given time frame. If a packet loss due to collision occurs, the receiver can send the information about the 'largest silent period' to its primary source. The source can then try to transmit at those specific times, so as to minimize the chances of further collisions. In essence, the source is trying to interleave its transmissions with background transmissions, while still keeping a constant data rate. We call this method Phase-offset collision avoidance.

Required Functionality

For both the Informed TCP-like and the Phase- offset methods, we need to add some extra functional- ity to both senders and receivers. Specifically, we need to be able to detect collisions, estimate the variation in the transmission times of packets and calculate the largest silent period between two consecutive packet ar- rivals.

Collision Detection

Determining collisions at the receiver is not as straightforward as just checking for corrupted pack- ets, since corruption can be due to both link losses and collisions. Also since the packet is corrupted we can- not always inspect it to determine its origin. There- fore, the only thing we can infer is that a loss oc- curred .Ideally, the receiver would know exactly when each source's packets arrived. If multiple packets were to arrive at the same time, the receiver could determine that a collision occurred and inform its source. The ideal solution can be approximated by adding a small amount of determinism to the protocol. In many ap- plications, the source will send a packet, and then set a timer to schedule the next transmission. If we deter- mine the next transmission time before sending the cur- rent packet, we can add this information in the payload, thus informing the receiver when to approximately expect the next packet. Since every source follows the same rules, the receiver can construct an arrival time- line for all the sources it can overhear. If the primary source and any background source(s) indicate that they will arrive at approximately the same time in the f ture, we can hypothesize that any corrupted packet that arrives at that time is the result of a collision. Figure 1 depicts this situation. A CSMA MAC includes a back off mechanism to avoid collisions. This mechanism can add a non- deterministic delay between the time at which the source passes the packet to the MAC layer and starts its timer and when the receiver receives the packet and builds its relative time offset. The source can send the variance together with the transmission interval to the receiver, to allow for a more accurate estimation of the next packet's arrival time. We dis- cuss packet transmission time and variance calculations in section 4.2.

The transmission interval and the transmission variance define a range over which the next packet may arrive. Collisions are now determined by the overlap of multiple

time ranges. Since packets can arrive at any time in the range, it is possible that though ranges over- lap a collision may not occur. Thus, the collision detection model cannot be used to accurately predict fu- ture collisions. It can only be used for post-facto collision detection.

Based on the above, a lost packet is classified as a collision-induced loss if, by the end of expected arrival time plus the transmission variance, the following has occurred:

•No data packet from the primary source has arrived.

•The arrival timeline showed an overlap between the primary source's arrival time and a back- ground source's arrival time.

•At least one corrupted packet was received.

We note that our collision detection mechanism is an estimator —it doesn't always guarantee a correct re- ply. If a source's packet does not arrive when expected, we can only say that it collided with some probability, since we are dealing with ranges as well as incomplete information. It may be possible that a detected collision was predicted to collide with another source, but was actually corrupted due to increased noise levels (link loss). Our estimator would then produce a false positive.

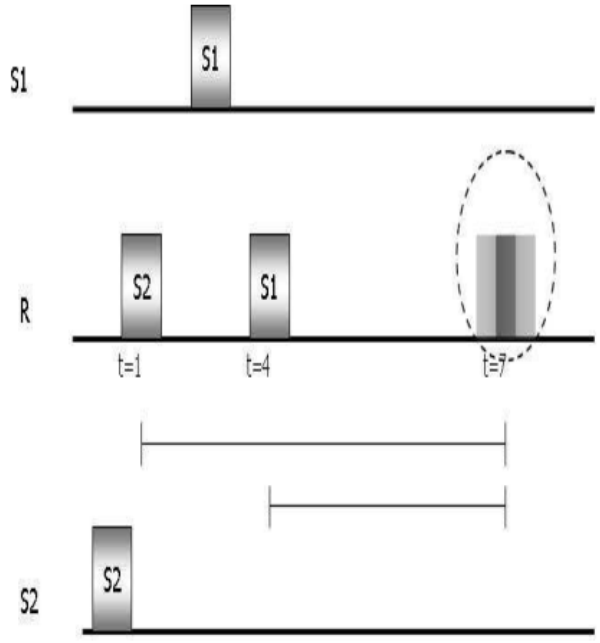


Figure 1. Collision detection at the receiver. Source S2 transmits at t = 1 and informs the receiver that the next transmission will be at t = 7. Source S1 trans- mits at t = 4 and informs the receiver that the next transmission will occur at t = 7. If at least one corrupted packet arrives at t = 7, the receiver infers that a collision occurred.

If no packet is received at the expected. time and no other source was known to transmit at that time, then the packet is assumed to have been lost due to radio propagation. However, if other sources were expected to transmit at that time, the packet's fate is unknown: it could have been lost due to radio propagation or a collision. If the collision occurs in the preamble or the start symbol, then the receiver will not be able to lock on to the sender and no packet will be received. This would lead to a false negative.

Transmission Time

When a packet arrives at the receiver, the transmis- sion time (which includes propagation delay and MAC transmission delay) is implicitly included. If we assume that there are no variations in the transmission time, then packets originating from a source every Tp sec- onds would be received at the receiver at Tp + Ttx seconds, where Ttx is the transmission time (mote radios have very small ranges, therefore propagation time is negligible).

In reality, the transmission time is not constant. The receiver can wait for a small amount of time after the expected for transmission time variations. This additional time, or 'guardband' can be pre-configured based on experimen- tal observations. But a statically assigned value can- not adapt to arbitrary network loads [7]. We thus turn again to TCP for a solution.

The source calculates the transmission time of each packet it sends, by subtracting the transmission start timestamp from the transmission end timestamp. This sample is then used to update the variance of the transmission time.

The calculation is the same as the one used to estimate the TCP retransmission timeout (RTO) [7]. The value of the variance is sent to the receiver together with the expected time of the next transmission.

The receiver now sets its expected reception timer to Tp + 4v, where v is the variance reported by the source, and Tp is the expected arrival time of the next packet.

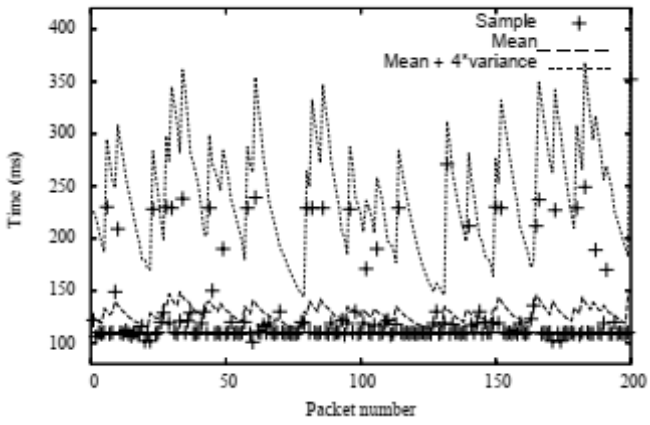


Figure 2. Transmission time variations at the source. The experiment consisted of two nodes. The primary source (node 1) transmits as fast as possible, while the background source (node 2) transmits every 530ms.

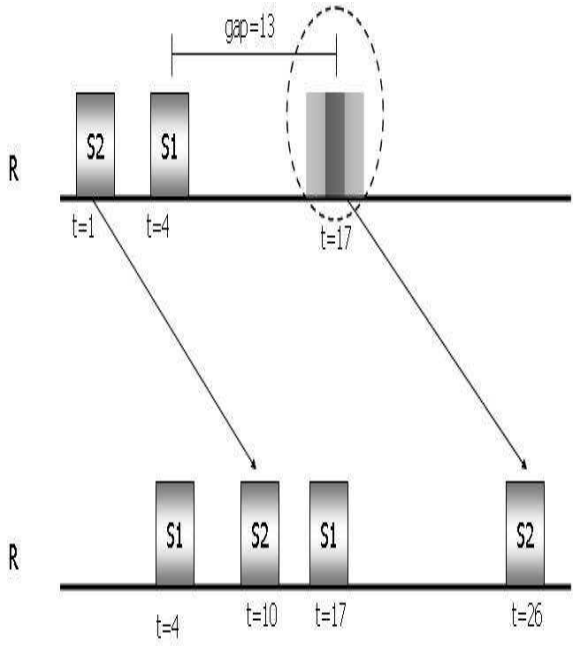


Figure 3. Changing the phase of a transmission while keeping the rate constant can lead to collision avoidance.

The link quality between the two nodes was approximately 60%.

Largest Silent Period Detection

Consider a scenario where two sources are transmit- ting at constant periods T1 and T2. We assume that channel utilization is low, so the aggregate rate is not high enough to saturate the channel. Hidden terminal collisions happen when the actual transmission times Ttx1 and Ttx2 are overlapping. Since the utilization is low, we can avoid the overlap (and thus collisions) while keeping the rates account corrupted packets. Even though those packets are application's perspective, they are still an indication that the channel is busy at that particular point in time, so constant, by adjusting the phase at which the source's The receiver can provide the source with the neces- sary information by calculating the largest time differ- ence between two consecutive packet arrivals, within a given time period. Since the entire system is clocked off of the source's transmission period, this is also the period used for the calculation of the time difference. We also take not useful from the theytransmissions happen. shouldn't be ignored. Figure 3 shows an example of the phase off- set calculation.

3. IMPLEMENTATION

We implemented our designs using Tiny OS [5], since motes are MOAP's target platform. Our application sends a series of packets from the source to the receiver in sequential order. Each packet contains a sequence number, the next transmission time and the calculated variance.

The receiver uses a bitmap to store successful packet receptions. Every time a packet arrives from the primary source, the receiver sets a timer based on the next expected transmission time plus any variation defined by the guard band.

A packet loss is discovered if no packet arrives by the end of transmission time. At this point the receiver scans the bitmap for the first missing packet. It then sends a NACK to its source. The source will retransmit requested packets before sending any new packets.

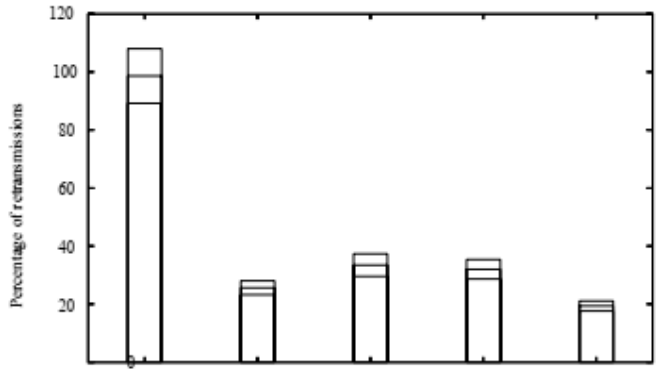


Figure 4. Percentage of retransmissions for each method, with two sources transmitting 200 packets. The single frequency phase offset method was very close to the theoretical minimum of 20%. The base case was the worst, incurring a penalty of almost 100%.

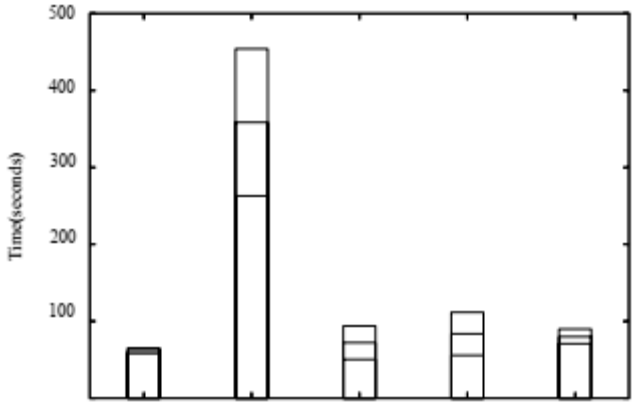


Figure5. Time required for the primary source to complete its transmission, with two sources sending 200 packets. The base case is the fastest. The phase off- set methods and informed-TCP are evenly matched, the latter slightly faster in some cases, due to its adaptive properties. The uninformed-TCP is approximately five times slower than the base case

Uninformed TCP-like Collision Avoidance

Our implementation of source-based collision avoidance follows the general design of TCP congestion avoidance. The initial transmission period of the primary source is 500ms. For each data packet that is successfully received, the source will additively increase its data rate, using the rate-based formula described in [7] with a = 10.
For each data packet that is lost at the receiver, the source will multiplicatively decrease its data rate. Since MOAP uses NACKs to notify the source of losses, they are used to trigger the multiplicative decrease. To avoid cases where synchronized sources stay synchronized even after the multiplicative decrease, a random floating-point number between 1.5 − 1.9 is used.

Informed TCP-like Collision Avoidance

The source will react much like it did in the uninformed TCP-like model. The initial transmission period is again 500ms. For each successful packet sent, the source will again additively increase its data rate. For each packet lost the source will now distinguish if the loss was due to link loss or collision. The source will multiplicatively decrease its data rate only when the receiver indicates the loss is due to collision but will take no action on link losses.

Phase-Offset Collision Avoidance

We again use the NACK mechanism to pass the information back to the source from the receiver. In the absence of NACK packets, the source implicitly assumes that everything is working properly, since no receivers complain. An arriving NACK will now include a time initially randomly selected from 500-2500ms. In the offset indicating the largest silent period time. When the source receives the NACK, it will set a timer for this particular offset. After that timer expires, it will resume its regular transmission, at its constant rate.

4. EVALUATION

In order to evaluate the performance of our algorithms, we used the Em Star framework [3] and Em TOS [4]. Em TOS allows development of NesC applications in the resource-rich environment of a 32-bit plat- form. Em Star includes a packet-level simulator which provides different channel models. For our simulations, we used an empirical channel model, collected from the ceiling array in our lab [2].

Two Sources, Single Receiver

We initially compared the performance of our algorithms using a simple, three-node setup. The first node was the primary source, the second was the receiver and the third one was the background source. The nodes were arranged in a line. The link quality be- tween nodes 1 and 2 (primary source and receiver) and nodes 2 and 3 was approximately 80%. The link quality between the two sources was approximately 50%. The sources in all cases sent 200 packets. The packet size was 150 bytes. All results were averaged over 10 experimental runs. Error bars on the figures represent 95% confidence intervals.
In both TCP-like cases, the background source transmits at a constant period that is

phase off- set case, we used two different setups. The

first one (phase-rand) was equivalent to the TCP-like cases. The second one (phase-single) had the back- ground source transmit at the same rate as the primary source (500ms), but with a random initial starting time. Intuitively, the second case would provide better results. The reason is that if the frequencies of the two sources are not integer multiples of each other, the largest silent period will not be constant; rather, it will keep shifting as one frequency drifts into another. If a collision occurs and is

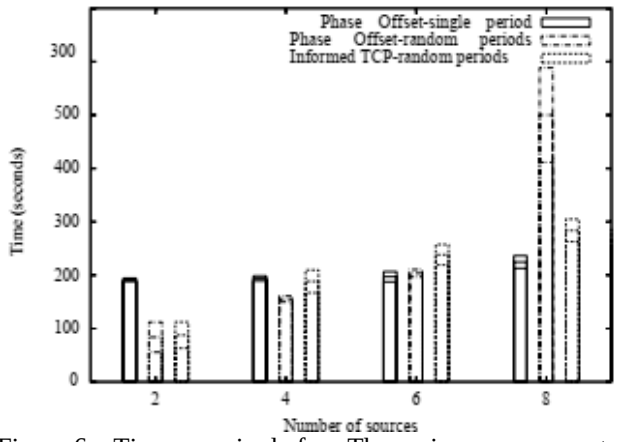
detected, there is little guarantee that the new phase offset will be collision free.

For this particular set of experiments, we also implemented a 'base' case, in which each source sent packets as fast as possible (back-to-back). We expect this to produce the fastest completion time, but also have the largest number of collision-induced retransmissions.

We first compare the different methods in terms of number of required retransmissions. As Figure 4 shows, the single-period phase offset was very close to 20%—the expected number of retransmission in the absence of collisions, since the link quality was 80%. The base case was the

worst, since operating at or close to the channel capacity leads to a large number of hidden teminal collisions. Collision avoidance schemes were thus able to reduce the number of retransmissions by al- most a factor of 8.

The reason why uninformed-TCP performed so well in reducing retransmissions can be seen in Figure 5. The completion time of uninformed-TCP was so large, that the background source, which transmits at a constant rate, had already finished its transmissions. The uninformed-TCP method, unable to distinguish be- tween collisions or link losses, was backing off on each NACK received.



Figure;6. Time required for The primary source to complete its transmission, versus number of total sources. Initially, informed-TCP and random-periods phase off- set have the best performance. However, random- periods phase offset scales poorly as the number of sources increases. The other two methods have good scaling

Consequently, for a significantly large period of time, only the primary source was (slowly) transmitting. All losses for that time were due to radio propagation, not collisions. The other methods had al- most equivalent results, with the exception of the base case which was the fastest.

By combining the results for latency and number of retransmissions, we can obtain the additional energy requirements that each method incurs. In our calculations, we used the values reported in [9] (20.000nAh to transmit a packet and 1.250nAh per millisecond for idle listening). From Figure 2 we see that the mini- mum transmission time per packet is approximately 120ms. We therefore used that value for our calculations and assumed that transmission times are constant. We also assumed that when the node was not transmitting, it was idle and thus used the 'idle listening' value for those periods of time. This however ignores the cost of packet reception which is not equal to the idle listen cost. In that sense, our results are more optimistic.

We calculated the total energy usage on the primary source node and then subtracted the energy needed to transmit 200 packets back-to-back the optimal energy usage for all cases. The results are shown in Figure 6. The single-period phase offset had the best perfomance, since it had the smallest number of retransmissions and was one of the fastest methods. Uninformed- TCP had the worst performance. Even though it was among the best in terms of additional retransmissions, it took so long to complete that the idle listen energy consumption had a large negative impact. Phase off- set and informed-TCP were able to reduce the additional energy penalty by almost 50% compared to the base case.

properties.

Multiple Sources, Single Receiver

The second set of experiments involved studying the behavior of the three better-performing methods—informed TCP and the two phase-offset variants—as the number of background sources increases. The placement of the initial three nodes, used in the previous set of the experiments remained the same. Extra sources were added in random locations within a 10-by-10 meter square. We run experiments with 2, 4, 6 and 8 sources.For this particular experimental setup, we changed the single-period phase offset slightly; the transmission period of all sources was now 800ms. With each packet transmission taking approximately 120ms, this variant of phase offset can theoretically accommodate up to seven non-overlapping sources. In practice, since we don't care about background source overlap (only overlaps with the primary source can cause important data loss on the receiver), more sources can be accommodated. The

random-period variant and the informed- TCP case had a transmission period of 500ms.

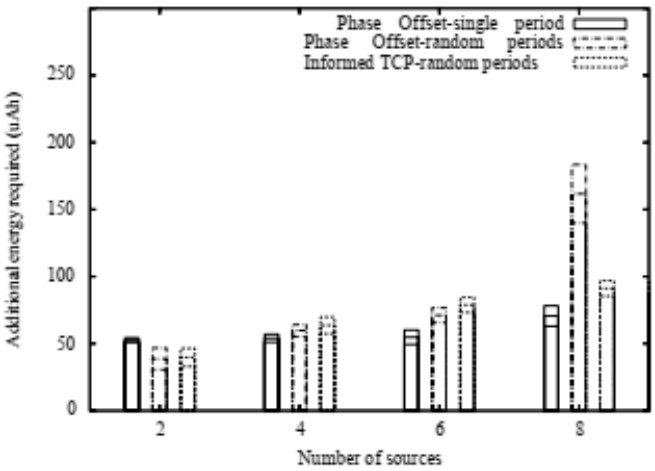


Figure 7. Additional energy requirements on the primary source, versus number of total sources. Single- period phase offset again gives the best performance, followed by informed TCP.

increases.Therefore, if one can design the system to pick the right frequency and make every node take the same frequency constant, phase offset is the best method to use.If a system does not have a commonly agreed upon frequency, or is highly dynamic, then the informed TCP method is preferred.

5. CONCLUSIONS AND FUTURE WORK

In resource-constrained wireless sensor networks, collisions can be a major cause of packet retransmissions and hence increased energy consumption. Traditional MAC-layer solutions solve the problem only to certain extent and have rather limited applicability. In this paper we take advantage of inherent multi-packet communication in our specific application of code dissemination to develop two application-based solutions to collision avoidance: a TCP-like method that uses collision detection as an input signal to adjust transmission rate, and a 'phase-offset' method that shifts packet transmission time within a given time frame to minimize collisions. Simulation results show that both of our solutions can significantly reduce the number of re-transmissions and energy usage

Figure 9. Additional energy requirements on the primary source, versus number of total sources. Single- period phase offset again gives the best performance, followed by informed TCP.

Our results indicate that if the system characteristics, such as the maximum number of concurrent sources, are known, one can pick a common transmision period to accommodate all the sources. In this case, the phase offset method is very effective at re- ducing collisions. In cases where sources transmit using different periods, the informed-TCP method, which can adapt to network dynamics, can be used to effectively reduce collisions.

As part of our ongoing effort, we plan to do several additional experiments that would provide further insight and confidence on the performance of our algorithms:

Running experiments with multiple sources and multiple receivers. Multiple receivers may lead to different feedbacks to each source, this is another interesting aspect that we need to investigate.

Running experiments with real hardware. Every simulator model, no matter how accurate can still not match reality. We therefore plan to run experiments with real hardware, using our ceiling array.

Further exploration of the design space. We are looking into combining the phase-offset method with the informed-TCP method by using the latter to adjust the common time frame period while using the former to shift each source's transmission time within the time frame.

Integrating the most appropriate solution in MOAP. Since MOAP was the driving application for our work, we plan on integrating our final collision avoidance solution into it to improve its performance.

The results, in terms of number of retransmissions, completion time and energy cost are shown in Figures 7, 8 and 9 respectively.

Again, as in the two- source case, the single-period phase offset gives the best results. However, the random-periods phase off- set now performs much worse, especially as the number of sources increases to eight. Even though the channel isn't necessarily over-utilized (since sources transmit at random periods), the 'frequency drift' effect is very pronounced. In effect, phase-offset cannot find a solution that will be valid for a long period of time.

Even though the single-period phase offset performed the best in all cases, we noticed a significant inrease in the number of retransmissions when the number of sources increases to eight from six.

This indicates that the channel is reaching capacity and that a further increase in the number of sources will be detrimental to the performance of the phase-offset method. Informed TCP performed adequately, degrading gracefully as the number of sources

REFERENCES

[1] H-c Lee, J-w Lee, J-h Hwang, H Yoe, in Future Generation Information Technology. A study on energy efficient MAC protocol of wireless sensor network for ubiquitous agriculture (Springer, Berlin Heidelberg, 2010), pp. 642–650.

[2] R Sokullu, E Demir,MAC layer protocols for linear wireless sensor networks :a survey. Commun. Special Issue on Ad-hoc Netw. 38(4), 393–422 (2014).

[3] S. Sendra, J. Lloret, M. Garcıa, and J. F. Toledo, "Power Saving and energy optimization techniques for wireless sensornetworks,"JournalofCommunicationsvol.6,no.6,pp. 439–459, 2011.

[4] O. Diallo, J. Rodrigues, M. Sene, and J. Lloret, "Distributed Database management techniques for wireless sensor networks," IEEE Transactions on Parallel and Distributed Systems,2013.

[5] L. M. L. Oliveira, J. J. P. C. Rodrigues, A. G. F. Elias, and B. B.Zarpeℓao, "Ubiquitous monitoring solution for Wireless Sensor Networks with push notifications and end-to-end connectivity, "Mobile Information systems, vol.10, no.1, pp.19–35,2014.

[6] O. Diallo, J. J. P. C. Rodrigues, and M. Sene, "Real-time data management on wireless sensor networks: a survey," JournalNetworkandComputerApplicationsvol.35,no.3,pp. 1013–1021,2012.

[7] O. Boyinbode, H. Le, and M. Takizawa, "A survey on clustering algorithms for wireless sensor networks, "International Journal of Space-Based and Situated Computing,vol.1,no.2,pp.130–136, 2011.

[8] N. Aslam, W. Phillips, W. Robertson, and S. Sivakumar, "Amulti-criterion optimization technique for energy efficient cluster formation in wireless sensor networks," Information Fusion, vol. 12, no. 3, pp. 202–212, 2011.

[9] D. Karaboga, S. Okdem, and C. Ozturk, "Cluster based wireless sensor network routing using artificial bee colony algorithm," Wireless Networks,vol.18,no.7,pp.847–860,2012.

[10] Naeimi, H.Ghafghazi, C.O.Chow, and H.Ishii, "Asurveyonthe taxonomy of cluster-based routing protocols for homogeneous wireless sensor networks," Sensor, vol.12, no.6, pp.7350–7409, 2012.

[11] J. Lloret, M. Garcia, D. Bri, and J. R. Diaz, "A cluster-based architecture to structure the topology of parallel wireless sensor networks," Sensors, vol.9, no.12, pp.1051310544,2009.

[12] M. Lehasini, H. Guyennet, and M. Feham, "Cluster-based energy-efficient k-coverage for wireless sensor networks," Network Protocols and Algorithms, vol.2, no.2, pp.89–106, 2010.

[13] R. Rajagopalan and P. K. Varshney, "Data aggregation techniques in sensor networks: a survey, "IEEE Communications Surveys,vol.6,no.4,pp.48–63,2006.

[14] K. Maraiya, K. Kant, and N. Gupta, "Wireless sensor network: are view on data aggregation," International Journal of Scientific & Engineering Research,vol.2,no.4,pp.1–6,2011.

[15] J. N. Al-Karaki, R. Ul-Mustafa, and A. E. Kamal, "Data aggregation and routing in Wireless Sensor Networks: optimal and heuristic algorithms," Computer Networks,vol.53,no.7,pp.945–960, 2009.

[16] H.O.Tan, I.Korpeoglu, and I.Stojmenovic, "Computing localized power-efficient data aggregation trees for sensor networks," IEEE Transactions on Parallel and Distributed Systems,vol.23,no.3,pp.489–500,2011.

[17] Q. Gao, Y. Zuo, J. Zhang, and X.-H. Peng, "Improving energy efficiency in a wireless sensor network by combining cooperative MIMO with data aggregation,"IEEE Transactions on Vehicular Technology, vol.59, no.8, pp.3956–3965, 2010.

[18] Wei, Y.Ling, B.Guo, B.Xiao, and A.V.Vasilakos, "Prediction-based data aggregation in wireless sensor networks: combining grey model and KalmanFilter," ComputerCommunications,vol.34, no.6,pp.793–802,2011.

[19] L. Xiang, J. Luo, and A. Vasilakos, "Compressed data aggregation for energy efficient wireless sensor networks," in Proceedings of the 8th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON '11), pp. 46–54, Salt Lake City, Utah, USA, June 2011.

[20] Y. Xua and J. Choi, "Spatial prediction with mobile sensor networks using gaussian processes with built-in gaussian markovrandom fields," Automatica, vol.48, no.8,pp.1735–1740,2012.

[21] J.-K. Min and C.-W. Chung, "EDGES: efficient data gathering in sensor networks using temporal and spatial

correlations," Journal of Systems and Software, vol.83, no.2,pp.271–282,2010.

[22] C.C.Hung,W.C.Peng,andW.C.Lee, "Energy-awareset-covering approaches for approximate data collection in wireless sensor networks," IEEE Transactions on Knowledge and Data Engineering, vol.24, no.11, pp.1993–1200, 2012.

[23] Y. Kotidis, "Snapshot queries: towards data-centric sensor networks," in Proceedings of the 21st International Conference on Data Engineering (ICDE '05), pp. 131–142, April 2005.

24 C. Liu, K. Wu, and J. Pei, "An energy-efficient data collection framework for wireless ensornet works by exploiting spatiotemporal correlation," IEEE Transactions on Parallel and Distributed systems, vol.18, no.7, pp.10101023,2007