Eindhoven University of Technology

Energy Efficient Embedded Systems

5KK07

Trickle–Scanning: An Energy Efficient Multichannel Communication Approach in Low Power Networks

Author:
Deepak Vedha Raj Sudhakar(0925172)
Shashidhar Bangalore Lakshmana (0926697)
Vaibhav Milind Kulkarni (0928138)

Supervisor:
Dr.Majid Nabi Najafabadi

January 28, 2015

Trickle–Scanning: An Energy Efficient Multichannel Communication Approach in Low Power Networks

Deepak Vedha Raj Sudhakar*, Shashidhar Bangalore Lakshmana†, Vaibhav Milind Kulkarni‡, Dr.Majid Nabi Najafabadi §
Department of Electronic Systems
Eindhoven University of Technology
Email: *d.v.r.sudhakar@student.tue.nl, †s.bangalore.lakshmana@student.tue.nl, ‡v.kulkarni@student.tue.nl §M.Nabi@tue.nl

***Abstract*—Adaptive channel hopping multichannel communication is an attractive solution to mitigate interference in the unlicensed ISM band (2.40 –2.48GHz) and provide reliable end to end packet delivery in low power 802.15.4 networks. Recently various standards viz. IEEE 802.15.4e (TSCH) [1], ISA 100.11a [2], Wireless HART [3], etc. have resorted to multichannel communication because of its benefits in crowded unlicensed bands. A major drawback of channel hopping techniques is the additional energy consumed in performing time synchronization, channel scanning, channel hopping and hopping pattern dissemination. These factors form the base of multichannel communication and leads to accurate channel hopping communication scheme. Implementing this scheme also leads to an increased code complexity further contributing to higher energy consumption. Energy efficient operation is an important aspect in sensor networks considering their long life cycles and small form factor. However, a systematic study evaluating the power consumption contribution by each factor is still missing. In this paper we experimentally analyze the portion of energy consumed by synchronization and channel scanning which contributes to energy overhead in multichannel communication. Further, we present an adaptive channel scanning approach to minimize the overhead contributed by channel scanning to blacklist certain channels in multichannel communication. We compare the energy consumption of periodic channel scanning approach shown by Peng Du et al [4] and our Trickle scanning approach which adjusts the scanning interval based on channel dynamics. By means of simulation, we show our approach "Trickle–Scanning" contributes to energy saving under different channel characteristics without affecting overall reliability of the WSNs.**

Keywords: Channel hopping, Low power networks, multichannel communication, IEEE 802.15.4e, Energy efficiency, Adaptive scanning.

I. INTRODUCTION

Today, majority of the wireless appliances used in household, industry and medical fields operate in the ISM band [5]. These devices need to coexist in the limited spectrum tolerating their mutual interference. There are several techniques to achieve the coexistence such as adjusting the transmission timing [6], forward error correction [7], adaptive routing [8] and the topic of interest of this paper, frequency agility [9]. In this technique the devices (sensor nodes) involved in communication, change their radio channel (operational frequency) periodically. Thus the communications will not suffer from the same interference leading to a stable and fair operation.

There have been several studies comparing different interference mitigation strategies with respect to latency and end to end packet delivery [10] . The interference mitigation problem is even grave for 802.15.4 compliant networks due to their low transmit powers, typically lying in the range of 0dBm to -25dBm. In comparison most of the RF devices operating in the ISM band transmit at much higher frequencies. IEEE 802.11 a/b/n (Wi-Fi) typically transmits at 20dBm peak power [11], IEEE 802.15.1 (Bluetooth) uses 4dBm peak transmit power [12], Cordless phones uses +60dBm peak transmission power [13]. 802.15.4 standard [14] specifies 16 usable channels lying in the range of 2.40 –2.48GHz. Each channel is 2MHz wide with a channel separation of 3MHz.

The interfering sources can occupy the channel in various ways. A narrow band interferer such as Bluetooth or a ZigBee device overlaps with one of the channels of 802.15.4. A wide band interferer such as Wi-Fi overlaps with 4-5 subsequent channels of 802.15.4. A sub band interferer such as wireless camera or cordless phone overlaps with a part of a channel of 802.15.4. Due to the complex nature of the interfering sources, moving away from the affected channel is one of the better approaches to overcome interference. This is evident by the fact that most of the new wireless standards adopting channel hopping scheme.

However, to achieve better reliability, multichannel communication trades off energy which is the most important factors in low power networking. The energy overhead is due to increased radio operation, CPU operation and code complexity. To achieve efficient channel hopping scheme, the nodes present in the network require tight time synchronization, this involves additional transmission and reception of time synchronization messages. Further, to find out the channels which are crowded, the nodes should perform periodic channel scanning to keep track of noisy channels. When a node decides to blacklist a channel, this information has to be disseminated to neighbouring nodes in the network contributing to power consumption. Implementation of all the above functionalities results in increased code complexity, leading to more energy usage.

In this paper, we evaluate each of the above factors with respect to energy consumption and code complexity. We present the energy share contributed by each factor resulting in a comparative analysis. Further, we propose a a novel

approach "Trickle–Scanning" to adapt the spectrum scanning rate depending on the environmental interference. We present a comparative analysis between energy consumption of periodic channel scanning approach shown by Peng Du et al [4] and our Trickle scanning approach which adjusts the scanning interval based on channel dynamics. We use Contiki OS an open source OS for Internet of Things [15] and Cooja simulator [16] for performing the simulations. Cooja is a cross layer simulator, which allows to perform experiments and edit operation of each layer of the OSI model.

Rest of the paper is structured as follows. In section II, we conduct a thorough literature survey to review various multichannel communication algorithms, comparative analysis of single channel and multichannel communication and existing studies regarding multichannel frameworks. In section III, we present the experimental setup, networks topologies and tools used to carry out experiments. In section IV we discuss our adaptive spectrum scanning approach "Trickle–Scanning". In section V, we discuss experimental results and key findings with respect to energy overhead due to synchronization and channel scanning. We perform comparative evaluation of periodic channel scanning shown by Peng Du et al [4] and our "Trickle-Scanning" approach .Finally conclusions are drawn in Section VI.

II. Related Work

The multi channel communication protocols are designed with wide perspective of objectives, few of the notable include analyzing impact of interference on the networks capacity, focus on reduction of jamming effects, reliable data dissemination, reduce contention in broadcast medium and many more. For instance, the work done by the authors in [17], shows different types of jamming on WSN hardware and the proposed solutions to enable continued communication despite ongoing attack, by channel hopping. Channel surfing mechanisms have been introduced such that the jammed nodes dynamically change their operating frequency. Typhoon [18] is a data dissemination protocol which switches among channels to reduce contention in broadcast medium, amplifying the benefits of spatial reuse. Multi-channel communication can also be used to overcome the congestion that can occur due to contention and interference in the network.

The multi channel protocols can be broadly classified based on channel assignment technique as: fixed channel, semi-dynamic channel and dynamic channel assignments.

In the fixed channel assignment, nodes are clustered into different frequencies, the protocol makes use of multi-channel communication to reduce the effects of interference due to co-existing networks. TMCP [19] is a tree-based multi-channel protocol for data collection applications. The goal is to partition the network into multiple subtrees while minimizing the intra-tree interference. This is achieved by assigning different channels to the nodes residing on different branches starting from the top to the bottom of the tree. However, it is difficult to have successful broadcasts due to the partitions, and contention inside the branches is not resolved since the nodes communicate on the same channel. One of the advantages of fixed channel assignment approaches is the ease of implementation since the dynamics due to channel switching and variations in the network topology are not considered. Hence fixed channel assignment multi-channel protocols have less energy overhead compared to single channel communication.

In protocols with semi-dynamic channel assignment approaches, nodes are assigned fixed channels but they can switch between channels in order to communicate with other nodes. Multi-channel clustered LMAC is an extension of single channel MAC [20] into the multi channel domain. LMAC is a scheduled-based protocol where nodes are assigned time slots to access the medium. TimeFrequency MAC (TFMAC) [21], is another TDMA based multi-channel MAC protocol. Slots are assigned by exchanging control messages during the contention slot at the beginning of each time frame. A multi-channel energy efficient MAC for wireless sensor networks [22] CMAC assumes a low-power wake-up radio besides a half-duplex transceiver available on the nodes. While the low-power wake up radio continuously samples the medium, the transceiver is turned on whenever it is needed for communication for energy efficiency purpose. CMAC uses a 2-hop coloring approach to assign channels to the receivers. Another multi-channel MAC protocol proposed for WSNs is Hybrid MAC(HyMAC) [23]. HyMAC also uses a combination of TDMA and FDMA. Time slots and frequencies are assigned according to Breadth First Search (BFS) order on a tree topology and the protocol is designed for converge cast type of traffic for data collection applications. Receiver-based channel assignment (RBCA) [24] where channels are statically assigned to the receivers (parents) so as to remove as many interfering links as possible in WSNs with a tree topology. Ramakrishnan et al. propose a multi channel MAC protocol (SMC MAC), based on a dedicated control channel approach [25]. The protocol uses one control channel and 8 other communication channels. Nodes communicate and negotiate for the channel to be used for message exchange on the control channel. Semi-dynamic channel assignment approaches needs a detailed coordination of channel switching between the senders and receivers in order to be on the same channel at the same time which adds to the energy overhead.

Protocols with dynamic channel assignment, Y-MAC [26] is the first example that uses dynamic channel assignment in WSNs. A combination of a dedicated control channel and a frequency hopping method is used, and is based on scheduled access. However, time slots are not assigned to the senders but to the receivers. MuChMAC [27] is another multi-channel MAC protocol that is based on the frequency-hopping approach. Similar to McMAC, nodes follow a parallel rendezvous using a pseudo-random number generator. Nodes calculate the channel number according to the current slot number and node ID. To support low-power operation, nodes wake up during only a small portion of a slot and sleep in the rest. Frequency hopping dynamic multi-channel communication protocols require tight time synchronization between the

nodes, hopping schedule determination, channel blacklisting which contributes to the energy overhead.

In single channel networks with heavy traffic, sensor nodes may suffer from a large number of collisions, interference, low delivery ratio, and long delivery latency. There are two benefits of using Multiple Channels: Multiple channels can increase the available network capacity, also can receive and transmit data in parallel. And finally, in multi channel communication, a node can transmit and receive data under less interference and collisions. In summary, they improve network throughput, minimize intra-network interference and avoid external interference.

Although multi-channel communication alleviates the interference problems in WSNs and increases the capacity, it contributes additional overhead in comparison to single channel communication. One of the most important issues in WSNs is the energy efficiency. Most of the multi-channel protocols discussed consider reliability as their primary goal where energy efficiency takes a backseat. Very few protocols such as Y-MAC, TMMAC consider energy efficiency as their primary goal. In this paper we quantitatively evaluate the energy overhead due to synchronization and channel scanning, which can provide insight on the energy contribution of these mechanisms to multichannel communication protocol designers. We propose a novel channel scanning "Trickle-Scanning" algorithm which can dynamically change the scanning interval based on channel characteristics. By this we avoid redundant channel scanning when the channel characteristics are stable.

III. EXPERIMENTAL SETUP

We use Contiki OS [28] and Cooja simulator [16] for implementation and carrying out simulations. In this section, we discuss the experimental setup, the libraries and tools used and the methodology followed to perform the experiments.

A. Network Setup and Topology

Since we perform a comparative evaluation, it is necessary to use a fixed topology and network parameters to conduct experimentation. We perform simulation using Tmote Sky [29] nodes. The network consists of 10 nodes. The network topology is depicted in Figure. 1. One root node (node id:1) which will send data packets to the leaf nodes in case of data dissemination and collect packets from all the leaf nodes in case of data collection. Four intermediate nodes (node id:2,3,4,5), which perform the packet forwarding operation from root nodes to leaf nodes and vice-versa. Five leaf nodes (node id: 6,7,8,9,10), which collect packets when root node is disseminating data and send packets to the root node in case of data collection.

We rely on RPL [30] (routing protocol for low power and lossy networks) to form the routing tree. Here, each node has a single parent to whom it forwards data or receive data from. A parent can have multiple children and it multicasts the data to all of them in case of data dissemination. Before executing the actual program, the RPL triggers the route discovery mechanism to form a directed acyclic graph (DAG) towards the root node. Depending on the rank of the nodes, each node selects its parent node and form the routing tree.

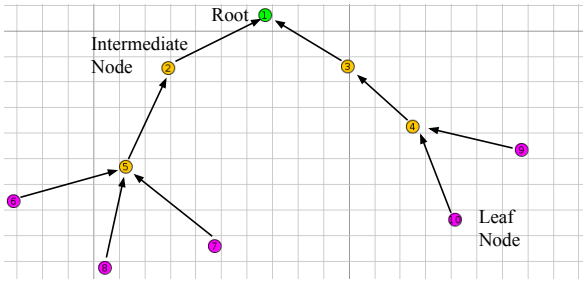

Figure. 1. Network topology in Cooja Simulator

B. Data Dissemination and Collection

Data dissemination and data collection are the two most prominent workloads in WSNs. We consider these two workloads as benchmark to evaluate the energy overhead of multi-channel communications. For data dissemination from the root node, we use the Trickle library [31] present in Contiki. The root node is programmed to send data packets of a fixed size after fixed durations. We use two packet sizes of 25 bytes and 50 bytes and transmit time interval of 10sec and 30sec respectively. Channel 26 is selected as a default channel for single channel communication and the packets are transmitted at the maximum transmit power of 0dBm. The root node broadcasts the data packets, the intermediate nodes forward them towards the leaf nodes. The trickle library controls the sending time instants and eliminates forwarding duplicate packets. We keep a track on the received packets at the leaf nodes to ensure successful packet delivery. In case of data collection we use UDP collect library [31] in Contiki. All the 5 leaf nodes send packets of a fixed size after fixed durations. We use two packet sizes of 25 bytes and 50 bytes and transmit time interval of 30sec and 20sec respectively The packets are forwarded the the leaf nodes towards the root node. In both the cases the simulation is run for 10 minutes keeping a track of the power consumption.

C. Energy consumption

The energy consumption is calculated by using the PowerTrace application and the Energest [32] module available in Contiki to monitor per component power consumption. The Energest module keeps track of the radio operation and counts the total timer ticks when the radio is transmitting, receiving the packets and the time elapsed in doing idle listening. It also keeps a track of the total time the CPU is in active and low power modes. The CPU current consumption while operating in active mode is 2.2mA, while in low power mode it is 0.00169mA, the radio consumes 18.05mA while doing packet transmission, the total radio receiver current consumption is 33.6mA, including idle listening . We assume a battery voltage

of 3 volts. The total energy consumption is calculated using the equation (1).

$$\frac{((CPU\times 2.2)+(LPM\times 0.00169)+(TX\times 18.05)+(RX\times 33.06))\times 3}{RTIMER\_ARCH\_SEC} \quad (1)$$

where, CPU corresponds to total rtimer ticks for which CPU is in active mode, LPM corresponds to total rtimer ticks for which CPU is in low power mode, TX corresponds to rtimer ticks for which radio is in transmit mode, RX corresponds to rtimer ticks for which radio is in receive mode including idle listening, RTIMER _ARCH _SEC corresponds to timer resolution total number of RTIMER ticks equivalent to one second. By dividing rtimer ticks by RTIMER _ARCH _SEC, we are able to obtain the time in seconds.

D. Time Synchronization

Device-to-device synchronization is necessary to maintain connection with neighbors in a slotframe-based network. Time is viewed as a series of consecutive superframes, each consisting of a configurable number of timeslots. Synchronisation is achieved by exchanging ASN-inserted advertisement (ADV) packets in dedicated ADV slots of each superframe.The chief approaches to perform time synchronization in 802.15.4e standard includes frame based synchronization [1] and acknowledgement based synchronization. In case of frame based synchronization the receiver nodes syncs with the transmitter node. On the other hand, in receiver based synchronization the transmitter syncs with the receiver. We choose frame based synchronization to sync all the nodes in the network with the root node in case of data dissemination as well as data collection. In Frame-based synchronization a node may synchronize its own network clock if it receives a frame from a time source neighbor. The receiver calculates the delta between expected time of frame arrival and its actual arrival time to use that information to adjust its own clock.

The root node timestamps the data packet while sending. We send the next packet at an interval of every T seconds. Thus the receiver knows the expected packet arrival time. If the packet is received at T$\pm\delta t$, we shift the clock at the receiver by t. The root node (node id:1) acts as time source for the forwarding nodes (node id: 2,3) as shown in Figure. 1. The forwarding nodes (node id: 2,3) acts as time source for forwarding nodes (node id: 4,5). The forwarding node (node id: 4) acts as a time source for leaf nodes (node id: 9,10). The forwarding node (node id: 5) acts as a time source for leaf nodes (node id: 6,7,8).

E. Channel Scanning

Wireless Hart standard and Bluetooth support the removal of certain channels from the channel hopping sequence, known as adaptive frequency hopping (AFH) [3] and Blacklisting [12], respectively, but provide no standardised implementations. To investigate the performance of blacklisting, [33] replays previously gathered channel hopping data traces with different blacklist sizes applied and finds an improved packet delivery rate (PDR). However the work is purely statistical and suggests no algorithm for blacklisting. An essential requirement for adaptive channel hopping is to identify the undesirable channels to be blacklisted. [34] demonstrates using Packet Delivery Ratio (PDR) as the metric for channel quality. Nevertheless, a relatively long sampling period (15 minutes) is required for each channel which makes this method inefficient in capturing changes in channel situations. Spectrum sensing, on the other hand, provides an alternative approach to channel condition assessment and is explored by Peng et.al [4].

In this approach, noise floor listening is conducted by accessing the Received Signal Strength Indicator (RSSI) from the radio chipset. Communications are periodically suspended to create quiet periods to acquire valid noise floor readings. A Blacklisting Manager (BLM) component is implemented in each node. It calculates the mean RSSI of a different channel during each quiet period and counts the number of the results that exceed a threshold of -87dBm, which is based on the findings in [35]. These counts are defined as the noise level indicators (NLI) of each channel. A blacklist has the form of a 16-bit mask and each bit corresponds to a channel. During each periodic blacklist updating, the associated bit of the channel with the highest NLI is set to false by BLM. Every time a node needs to hop to a new channel, its BLM first checks the channel number against the mask. If it is not blacklisted then operations continue as normal; when the channel turns out blacklisted, BLM randomly generates a new channel number and repeats the process until an allowed channel is acquired.

We evaluate the energy consumed, by implementing a spectrum scanner which scans all the 16 channels in the ISM band before transmitting every packet periodically in the channel scanning slot, where all the communications are put to silent. We record the RSSI value while scanning and compare against a set threshold value (-87dBm for experimentation). If the RSSI value exceeds the threshold, we consider the channel as busy and blacklist it from the set of available channels. The channel blacklist information is broad casted to all the 1 hop neighbors. This ensures that all the neighbors know the updated set of available channels.

IV. TRICKLE SCANNING

In this section we discuss our approach "Trickle-Scanning". In this approach, we adapt the spectrum scanning rate according to the RSSI trend on the channels. The channel scanning interval is initialized to minimum scanning interval. If we notice no channel activity on any of the channels and RSSI samples are within a acceptable threshold (-87dbm) we count the number of occurrences. We wait until this count exceeds a threshold to increase the scanning interval . On the other hand, if we observe that the RSSI readings are greater than acceptable threshold (-87dbm), we set the scanning interval to the minimum possible value. This leads to low spectrum scanning rate if there is no interference activity on the available channels and a high spectrum scanning rate if interference is observed on any of the available channels. Thus by saving unnecessary spectrum scanning runs we gain energy efficiency.

Our approach is depicted in Algorithm 1. Our approach provides flexibility by allowing the designers to calibrate the parameters based on the environment. We make use of parameter noisefactor to determine the channel dynamics. The channel scanning interval is initialized to minimum scanning interval I_{MIN} and *noisefactor* is set to 0. If we notice no channel activity on any of the channels and RSSI samples are within interval [-100dbm , (*RSSIMIN* = -87dbm)] we decrement the noise factor by factor of 1. We wait until this noise factor exceeds a threshold *NF_MAX* to increase the scanning interval. The number of times the scanning interval can be increased is limited by factor *IMAX*. On the other hand, if we observe that the RSSI readings are greater than acceptable threshold (*RSSIMIN* = -87dbm), we set the scanning interval to the minimum possible value I_{MIN} and noisefactor is set to *NF_MAX*. The list of parameters are introduced below:
I_{MIN}:= Minimum scanning interval (choosen based on coherence time of the channel).
IMAX:= Count limiting the maximum possible scanning interval.
RSSI_DELTAMAX := Maximum difference in RSSI samples for which noisefactor will not be decremented.
RSSI_SAMPLES := Total number of samples considered for calculating average RSSI of channel.
RSSIMIN:= Minimum acceptable threshold above which channel scanning interval will be set to I_{MIN}.
NF_MAX := Total number of counts for which RSSI of all the channels must be lesser than *RSSIMIN* and *rssi_delta* must be lesser than *RSSI_DELTAMAX*, provided noisefactor is initially zero.
CHANNEL_MAX := Maximum channel number that has to be scanned.
CHANNEL_MIN := Minimum channel number that has to be scanned.

Algorithm 1 Trickle Scanning algorithm

$I \leftarrow I_{\text{MIN}}$
$channel \leftarrow CHANNEL\_MIN$
$channel\_interval \leftarrow (CHANNEL\_MAX - CHANNEL\_MIN)$
for $m \leftarrow 0$ *to* $channel\_interval$ **do**
  $noisefactor[m] \leftarrow 0$
  $rssi\_delta[m] \leftarrow 0$
  $rssi\_ch[m] \leftarrow -100$
end for
for every $scanninginterval\ I$ **do**
  $cca \leftarrow 0$
  $cnt \leftarrow 0$
  for $i \leftarrow 0$ *to* $channel\_interval$ **do**
    $rssi \leftarrow 0$
    $rssi\_old \leftarrow 0$
    $cc2420\_set\_channel(channel)$
    $clock\_delay(200)$
    for $j \leftarrow 0$ *to* $RSSI\_SAMPLES$ **do**
      $rssi \leftarrow rssi + cc2420\_rssi()$
      $wait(20)$
    end for
    $rssi\_ch[i] \leftarrow (rssi/RSSI\_SAMPLES)$
    $rssi\_delta[i] \leftarrow rssi\_ch[i] - rssi\_old$
    if $((rssi\_ch[i] > RSSIMIN) \wedge (\neg(noisefactor[i] >= NF\_MAX)))$ **then**
      $noisefactor[i] = NF\_MAX$
    else if $((rssi\_ch[i] < RSSIMIN) \wedge (\neg(noisefactor[i] <= -NF\_MAX)) \wedge (rssi\_delta[i] < RSSIDELTA\_MAX))$ **then**
      $noisefactor[i] = noisefactor[i] - 1$
    end if
    if $noisefactor[i] == NF\_MAX$ **then**
      $cca++$
    else if $noisefactor[i] == -NF\_MAX$ **then**
      $cnt++$
    end if
    $channel++$
  end for
  if $(cca > 0)$ **then**
    $(I \leftarrow I_{\text{MIN}})$
  end if
  if $(cnt == channel\_interval) \wedge (I <= I_{\text{MIN}} \times IMAX)$ **then**
    $I = I + I_{\text{MIN}}$
  end if
end for

V. Simulation Results

In this section, we determine the energy overhead due to Channel Scanning and Synchronization by means of simulation. We also compare the energy overhead due to our approach Trickle Scanning and Periodic Scanning shown in [4]. We use Contiki OS [28] and Cooja simulator [16] for implementation and carrying out simulations. The network setup and topology and the types of workload to perform comparative evaluation are discussed in section III.

A. Synchronization

We perform experiments with synchronization and without network synchronization for data dissemination workload to evaluate the energy overhead for different packet sizes and synchronization interval. We compute the difference in the energy due to synchronization at different time intervals. We make use of the Trickle library [31] present in Contiki to perform data dissemination from the root node. In experiment 1, we set up the root node to disseminate data of packet size 25 bytes at an interval T_data = 10sec and in addition

root nodes send synchronization packets with an interval of T_sync = 10sec. We repeat the experiment by switching off the synchronization, to evaluate the energy overhead. Figure 2 shows the energy overhead in the network to perform data dissemination with synchronization (corresponding to experiment 1)in comparison to performing the same workload without synchronization. The energy overhead increases linearly at rate of 2.5 mJ/sec. The slope is identified using least squares fitting method. Since synchronization involves exchange of additional synchronization packet between the nodes in addition to data, the energy overhead due to Transmit and Receive operation of radio contributes to more overhead than CPU operation as the time increases.

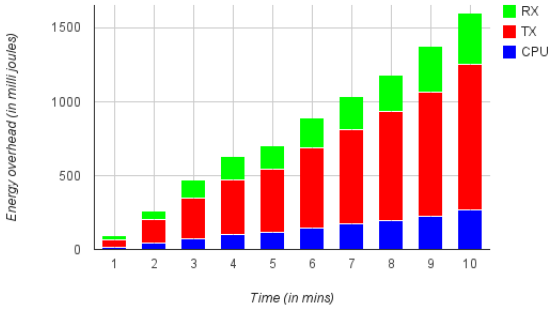

Figure. 2. Energy Overhead Vs Time, Workload = Data dissemination, Packet size = 25 bytes, T_data = 10sec, T_sync = 10sec

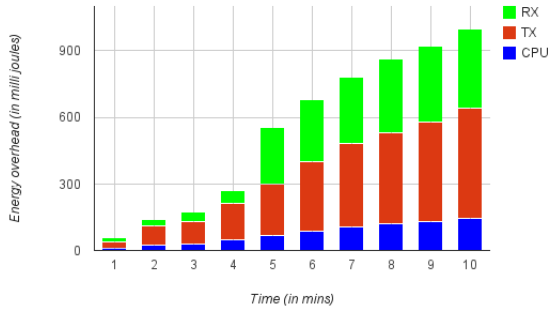

Figure. 3. Energy Overhead Vs Time, Workload = Data dissemination, Packet size = 50 bytes, T_data = 30sec, T_sync = 20sec

In experiment 2, the root node disseminates data of packet size 50 bytes at an interval T_data = 30sec and in addition root nodes send synchronization packets with an interval of T_sync = 20sec. We repeat the experiment by switching off the synchronization, to evaluate the energy overhead. Figure 3 shows the additional energy overhead in the network to perform data dissemination with synchronization (corresponding to experiment 2) in comparison to performing the same workload without synchronization. The energy overhead increases linearly at rate of 1.42 mJ/sec. As the synchronization interval decreases the energy overhead increases, as the nodes exchange synchronization packets more frequently costing more energy.

B. Channel Scanning

We perform experiments with periodic channel scanning and without channel scanning for data collection workload to evaluate the energy overhead for different packet sizes and scanning interval. Both the experiments are performed in interference free environment. We compute the difference in the energy due to channel scanning at different time intervals. We make use of UDP collect library [31] in Contiki to perform data collection from the leaf nodes to the root node. In experiment 3, the leaf nodes sends data of packet size 25 bytes at an interval T_data = 30sec and in addition all nodes perform channel scanning with an interval of T_scan = 20sec.

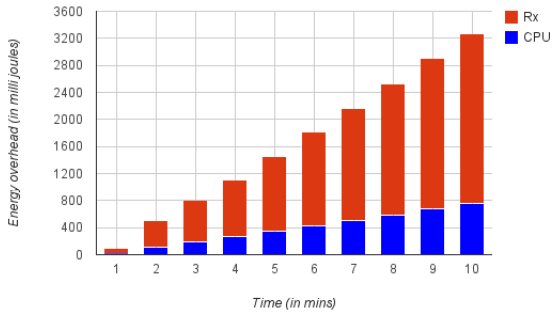

Figure. 4. Energy Overhead Vs Time, Workload = Data collection, Packet size = 25 bytes, T_data = 30sec, T_scan = 20sec

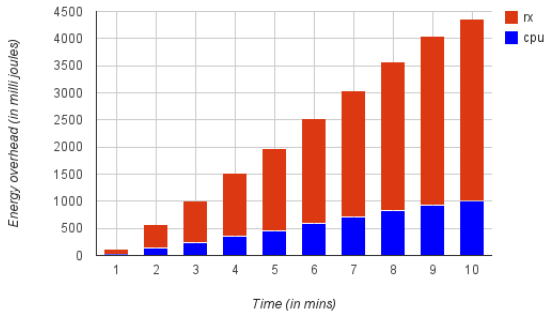

Figure. 5. Energy Overhead Vs Time, Workload = Data collection, Packet size = 50 bytes, T_data = 20sec, T_scan = 15sec

We repeat the experiment by switching off the channel scanning, to evaluate the energy overhead. Figure 4 shows the energy overhead in the network to perform data collection with periodic channel scanning (corresponding to experiment 3) in comparison to performing the same workload without channel scanning. The graph shows the overhead due to

CPU and Radio receive energy consumption only, as the difference in the transmit operation of radio is negligible. The energy overhead increases linearly at rate of 5.34 mJ/sec. Since the communications are periodically suspended and all nodes listen for noise during their scanning slot, the energy consumption due to radio in receive mode increases.

In experiment 4, the leaf nodes sends data of packet size 50 bytes at an interval T_data = 20sec and in addition all nodes perform channel scanning with an interval of T_scan = 15sec. We repeat the experiment by switching off the channel scanning, to evaluate the energy overhead. Figure 5 shows the energy overhead in the network to perform data collection with periodic channel scanning (corresponding to experiment 4) in comparison to performing the same workload without channel scanning. The energy overhead increases linearly at rate of 7.2 mJ/sec. As the scanning interval decreases the energy overhead increases, since nodes spend more time in noise floor listening.

C. Trickle Scanning Vs Periodic Scanning

We perform experiments with periodic channel scanning and Trickle scanning for data dissemination workload to evaluate the energy overhead for different channel interference pattern. We compute the difference in the energy due to periodic channel scanning and trickle scanning at different time intervals. We introduce microwave interference [13] in the channel 23 at distance of 1m from node 9 in topology Figure. 1 with a transmit power of -50 dbm across different time intervals. For interference pattern 1, we introduce the interference at the start of the network setup 0 to 2 minutes. For interference pattern 2, we introduce the interference at time 4 to 6 minutes, since the start of the network. For interference pattern 3, we introduce the interference at time 8 to 10 minutes, since the start of the network. A channel is stable when the noise floor readings (RSSI) does not fluctuate when the communications are periodically suspended to create quiet periods. A stable channel is one which is interference free.

In experiment 5, the root nodes sends data of packet size 25 bytes at an interval T_data = 10sec and in addition all nodes perform Trickle scanning with parameters ($I\_min = 7sec, IMAX = 5, RSSIMIN = -87dbm, NFMAX = 10, CHANNEL\_MIN = 11, CHANNEL\_MAX = 26, RSSI\_SAMPLES = 5, RSSI\_DELTAMAX = 10$) for interference pattern 1, pattern 2, pattern 3 and with no interference. We repeat the experiment by switching off the Trickle scanning and perform periodic scanning with an interval of T_scan = 7sec, to evaluate the energy overhead. Figure 6 shows the energy overhead in the network to perform data dissemination with periodic channel scanning (corresponding to experiment 4) in comparison to performing the same workload without Trickle scanning. Trickle scanning in general shows energy improvement under different channel characteristics over periodic channel scanning. When the environment is interference free, Trickle scanning shows the maximum energy improvement of 6800 mJ at the end of 10 minutes since the channel is stable for a long time, the scanning interval is increased from 7 seconds to a maximum of 35 seconds. For interference pattern 2, the energy improvement is minimal since channel is more dynamic and it does not allow the scanning interval to increase. The interference pattern 1 and 3 as energy improvement closer to without interference, since in both cases the channel is stable for a longer duration of 8 minutes in comparison to 4 minutes of pattern 1. Trickle scanning shows energy improvement when the channel is stable without interference. When the channel is dynamic the trickle scanning reduces the scanning interval behaving like periodic scanning hence the energy improvement is not much.

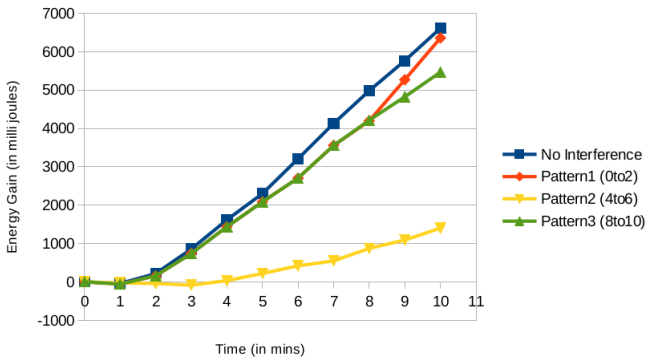

Figure. 6. Energy Savings(mJ) Vs Time (min), Workload = Data dissemination, Packet size = 25 bytes, T_data = 10sec, I_min = 7sec, IMAX = 5, RSSIMIN = -87dbm, NFMAX = 10, CHANNEL_MIN = 11, CHANNEL_MAX = 26, RSSI_SAMPLES = 5, RSSI_DELTAMAX =10 for with interference pattern1, pattern2, pattern3 and with no interference

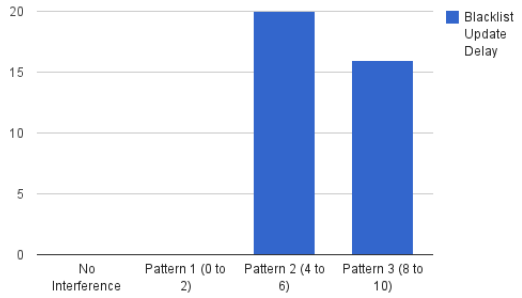

Figure. 7. Blacklist Update delay (sec) Vs Interference pattern, Workload = Data collection, Packet size = 25 bytes, T_data = 10sec, T_scan = 7sec for with interference pattern1, pattern2, pattern3 and with no interference

The figure 7 shows the delay in updating blacklist channels by Trickle scanning in comparison to periodic scanning for different interference patterns. With interference pattern 1, there is no delay introduced, since the interference happens at the start of the network setup all the nodes are set to minimum scanning interval at the start of the network setup. However for interference pattern 2 and pattern 3, Trickle scanning introduces a delay in blacklist channel under the interference influence since the channel was stable prior to the interference

the scanning intervals were increased. However this delay has minimal impact in the packet delivery ratio and does not affect the overall reliability of the WSNs when the Trickle scanning parameters are carefully calibrated based on the deployment environment.

VI. Conclusion

We have experimentally analyzed the portion of energy consumed by synchronization and channel scanning which contributes to energy overhead in multichannel communication. To perform the same workload we evaluate the energy overhead due to synchronization and channel scanning. We show that the channel scanning contributes to more energy overhead in comparison to synchronization. Further, we present an adaptive channel scanning approach to minimize the overhead contributed by channel scanning to blacklist certain channels in multichannel communication. We compare the energy consumption of periodic channel scanning approach shown by Peng Du et al [4] and our Trickle scanning approach which adjusts the scanning interval based on channel dynamics. By means of simulation, we show our approach "Trickle–Scanning" contributes to energy saving under different channel characteristics without affecting overall reliability of the WSNs. As a future work, we also plan to study the energy overhead in performing channel hopping and hopping pattern dissemination among the nodes in the network.

References

[1] "802.15.4.e." [Online]. Available: http://standards.ieee.org/findstds/standard/802.15.4e-2012.html

[2] "Isa 100.11a." [Online]. Available: http://www.isa100wci.org/

[3] "Wirelesshart." [Online]. Available: http://www.nivis.com/technology/WirelessHART.php

[4] P. Du and G. Roussos, "Adaptive channel hopping for wireless sensor network," *International Conference on Selected Topics in Mobile and Wireless Networking*, pp. 19–23, Oct. 2011.

[5] "Unlicenced ism band." [Online]. Available: http://www.afar.net/tutorials/fcc-rules

[6] N. Baccour, "Radio link quality estimation in wireless sensor networks: A survey," *ACM Trans. Sen. Netw. 8 Article 34*, Sep. 2012.

[7] C. Liang, N. Priyantha, J. Liu, and A. Terzis., "Surviving wi-fi interference in low power zigbee networks," *In ACM SenSys*.

[8] Q. D. Ho, T. N. Tran, and G.Rajalingham, "A distributed and adaptive routing protocol designed for wireless sensor networks deployed in clinical environments," *In Wireless Communications and Networking Conference*, pp. 2746–2750, Apr. 2012.

[9] Gonga.A, Landsiedel.O, Soldati.P, and Johansson.M, "Revisiting multi-channel communication to mitigate interference and link dynamics in wireless sensor networks," *In Distributed Computing in Sensor Systems*, pp. 186–193, May 2012.

[10] X. Zheng, Z. Cao, J. Wang, Y. He, and Y. Li, "Zisense: towards interference resilient duty cycling in wireless sensor networks," *In Proceedings of the 12th ACM Conference on Embedded Network Sensor Systems (SenSys '14)*.

[11] "802.11n." [Online]. Available: http://standards.ieee.org/findstds/standard/802.11n-2009.html

[12] "Bluetooth core specifications version 4.0, sig bluetooth std."

[13] A. Hithnawi, H. Shafagh, and S. Duquennoy, "Understanding the impact of cross technology interference on ieee 802.15.4," *WiNTECH '14 Proceedings of the 9th ACM international workshop on Wireless network testbeds, experimental evaluation and characterization*.

[14] "Ieee 802.15.4 standard." [Online]. Available: http://standards.ieee.org/about/get/802/802.15.html.

[15] "Iot." [Online]. Available: http://www.iot-a.eu/public

[16] Osterlind.F, Dunkels.A, Eriksson.J, Finne.N, and Voigt.T, "Cross-level sensor network simulation with cooja," *Local Computer Networks, Proceedings 2006 31st IEEE Conference on*, pp. 641–648, Nov. 2006.

[17] A. D. Wood, J. A. Stankovic, and G. Zhou, "Deejam: Defeating energy-efficient jamming in ieee 802.15.4-based wireless networks," *SECON 07: Proceedings of the 4th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks*.

[18] C. Liang, R. Musaloiu, and A. Terzis, "Typhoon: A reliable data dissemination protocol for wireless sensor networks," *EWSN 2008: Proceedings of the 5th European Conference on Wireless Sensor Networks*.

[19] Y. Wu, J. Stankovic, T. He, and S. Lin, "Realistic and efficient multi-channel communications in wireless sensor networks," *Infocom 08: Proceedings of the 27th IEEE International Conference on Computer Communications*.

[20] O.D.Incel, S. Dulman, and P. Jansen, "Multi-channel support for dense wireless sensor networking," *Proceedings of the First European Conference on Smart Sensing and Context, EuroSSC Lecture Notes in Computer Science, vol. 4272*.

[21] D.G.L.Jovanovic and Milica.D, "Tfmac: Multi-channel mac protocol for wireless sensor networks," *8th International Conference on Telecommunications in Modern Satellite, Cable and Broadcasting Services*.

[22] K. Chowdhury, N. Nandiraju, D. Cavalcanti, and D. Agrawal, "Cmac a multi-channel energy efficient mac for wireless sensor networks," *Proceedings of IEEE Wireless Comminication and Networking Conference (WCNC)*.

[23] S. Mastooreh, S. Hamed, and K. Antonis, "Hymac:hybrid tdma/fdma medium access control protocol for wireless sensor networks," *PIMRC 2007: The proceedings of the 18th IEEE Personal, Indoor and Mobile Radio Communications Symposium*.

[24] O. Incel and B. Krishnamachari, "Enhancing the data collection rate of tree-based aggregation in wireless sensor networks," *SECON 08: Proceedings of the 5th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks*.

[25] M. Ramakrishnan and P. Ranjan, "Multi channel mac for wireless sensor networks," *International Journal of Computer Networks and Communications*.

[26] Y. Kim, H. Shin, and H. Cha, "Y-mac: An energy-efficient multi-channel mac protocol for dense wireless sensor networks," *IPSN 08: Proceedings of the 7th International Conference on Information Processing in Sensor Networks*.

[27] J. Borms, K. Steenhaut, and B. Lemmens, "Low-overhead dynamic multi channel mac for wireless sensor networks," *EWSN 2010: Proceedings of the 7th European Conference on Wireless Sensor Networks*.

[28] Dunkels.A, Gronvall.B, and Voigt.T, "Contiki - a lightweight and flexible operating system for tiny networked sensors," *29th Annual IEEE International Conference on Local Computer Networks*, pp. 455–462, Nov. 2004.

[29] "Tmote sky datasheet." [Online]. Available: www.eecs.harvard.edu/konrad/projects/tmote-sky-datasheet.pdf

[30] ncillotti.E, Bruno.R, and Conti.M, "Reliable data delivery with the ietf routing protocol for low-power and lossy networks," *Industrial Informatics, IEEE Transactions on , vol.10, no.3,*, pp. 1864–1877, Aug. 2014.

[31] "Contiki library,url = http://contiki-os.blogspot.ch/."

[32] A. Dunkels, J. Eriksson, N. Finne, and N. Tsiftes, "Powertrace: Network-level power profiling for low-power wireless networks," *SICS Technical Report*, Mar. 2011.

[33] T. W. et al, "Reliability through frequency diversity: why channel hopping makes sense," *Proc. 6th ACM Symp. Performance evaluation of wireless ad hoc, sensor, and ubiquitous networks. ACM*.

[34] B. K. et al, "Feasibility analysis of controller design for adaptive channel hopping," *Proc. 4th Int. ICST Conf. Performance Evaluation Methodologies and Tools. ICST*.

[35] K. Srinivasan and P. Levis, "Rssi is under appreciated," *Proc. 3rd Workshop on Embedded Networked Sensors (EmNets*.