IEICE TRANS. COMMUN., VOL.E101–B, NO.3 MARCH 2018

PAPER

Zone-Based Energy Aware Data Collection Protocol for WSNs

Alberto GALLEGOS[†a)], ***Nonmember***, **Taku NOGUCHI**[††], **Tomoko IZUMI**[††], ***and*** **Yoshio NAKATANI**[††], ***Members***

SUMMARY In this paper we propose the Zone-based Energy Aware data coLlection (ZEAL) protocol. ZEAL is designed to be used in agricultural applications for wireless sensor networks. In these type of applications, all data is often routed to a single point (named "sink" in sensor networks). The overuse of the same routes quickly depletes the energy of the nodes closer to the sink. In order to minimize this problem, ZEAL automatically creates zones (groups of nodes) independent from each other based on the trajectory of one or more mobile sinks. In this approach the sinks collects data queued in sub-sinks in each zone. Unlike existing protocols, ZEAL accomplish its routing tasks without using GPS modules for location awareness or synchronization mechanisms. Additionally, ZEAL provides an energy saving mechanism on the network layer that puts zones to sleep when there are no mobile sinks nearby. To evaluate ZEAL, it is compared with the Maximum Amount Shortest Path (MASP) protocol. Our simulations using the ns-3 network simulator show that ZEAL is able to collect a larger number of packets with significantly less energy in the same amount of time.

key words: *wireless sensor networks, protocols, routing protocols, urban farming, precision agriculture, ns-3 simulator*

1. Introduction

According to the latest revision of the United Nations population prospects, the world population is projected to grow by 34 percent from 6.8 billion today to 9.1 billion in 2050 [1]. 70 percent of this population is expected to be urban. This will not only put pressure on the already scarce agricultural resources, but also aggravate the problem with the decrease of farming land. Farmers will have to make use of technologies to respond to these changes, producing more with less. Furthermore, it is estimated that as much as 25 to 50 percent of all food produced is wasted due to our inadequate distribution and storage systems [2]. In order to mitigate some of these problems, people living in urban settlements are required to be involved in the food production process. For example, by cultivating food in or around towns or cities (urban farming). Urban farming has grown in popularity in recent years with both private [3], [4] and public [5] initiatives around the globe. In highly populated cities, rooftops have served as the most common spaces used in urban farming but farms are not limited to these spaces. In fact, urban farms can take place in spaces as small as building balconies. Precision agriculture is a farming management concept based on observing and monitoring plants conditions, nutrients, and keeping track of statistics to boost productivity. Wireless Sensor Networks (WSN) used in Precision agriculture systems are frequently too "simplistic" from the point of view of routing design. These systems often involve 1-hop routing designs or cluster tree patterns [6]–[8]. One problem with these designs is the requirement of a manual positioning of the cluster heads which serve as a data collection points. Ideally, these must be positioned in range of each other and/or in the center of their corresponding cluster. This approach limits the network topology options and puts stress on some key network conjunctions. Moreover, cluster head nodes in this type of design usually need extra energy sources to support the additional required radio range and routing. In this paper, we describe in detail the Zone-based Energy Aware Data coLlection (ZEAL) routing protocol. ZEAL is a WSN protocol designed for but not limited to precision agriculture. It is able to overcome some limitations of traditional WSN protocols used in precision agriculture (central tree clustering based design and unsupported mobile data collection). For instance, ZEAL can be used in vertical farming [9], one of the world governments' efforts to achieve food sustainability in cities around the world (Fig. 1).

Additionally, in this paper, we propose a new communication time slot assignment algorithm called SelectiveTA to provide asynchronous communication between sensors and a mobile sink. Unlike existing protocols that use mobile sinks with constrained paths for data collection, ZEAL is a equipped with an energy saving mechanism implemented in the network layer during the data collection process. Building real implementations of wireless sensor nodes can be costly and time consuming. Using simulations allows for isolation of aspects of the developing process to focus on specific problems. The ns-3 simulator [10] is a popular discrete event network simulator used to evaluate a wide range of network aspects. Its direct code execution module (DCE) provides facilities to use the real Linux networking stack and network applications [11]. In other words, its emulation capabilities grant the possibility of reusing the same source code in testbeds (implementations of real physical wireless nodes) at no additional cost. For this reason, in this paper, we conduct a performance evaluation using ns-3. The paper

Manuscript received April 3, 2017.
Manuscript revised July 7, 2017.
Manuscript publicized August 28, 2017.

[†]The author is with the Graduate School of Information Science and Engineering, Ritsumeikan University, Kusatsu-shi, 525-8577 Japan.
[††]The authors are with the College of Information Science and Engineering, Ritsumeikan University, Kusatsu-shi, 525-8577 Japan.
a) E-mail: alramonet@yahoo.com
DOI: 10.1587/transcom.2017EBP3133

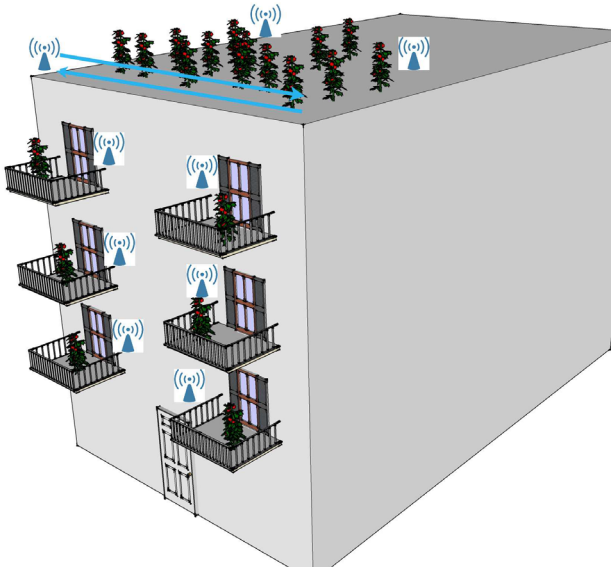


Fig. 1 Urban farm using ZEAL protocol. Information can be collected using a mobile sink at the top of the building.

is organized as follows:

Section 2 gives a brief description of related WSN protocols. Section 3 contains details of the classification and operation of the ZEAL routing protocol. We provide a detailed description of the communication time slot assignment, routing and duty cycling mechanisms used in ZEAL. Section 4 contains our evaluation of the protocol followed by our conclusions in Sect. 5.

2. Related Work

It is not uncommon to use Wireless Sensor Networks (WSN) in precision agriculture systems in the literature. However, optimization and development of such systems focus mainly on applications (irrigation, temperature control, status updates, etc). Because of its simple design, in precision agriculture the protocol Low Energy Adaptive Clustering Hierarchy (LEACH) [12] is arguably, the most common choice. LEACH is a clustering protocol that make use of a randomized rotation of the cluster heads to balance the energy consumption in the network. Its principal drawbacks (single hop routing and static cluster head selection and rotation) are addressed in multiple variants of the protocol such as HSEP [13], HEED [14], HEER [15] to mention a few. LEACH uses a probabilistic cluster head selection while the variants introduce additional parameters to select and rotate the cluster heads. Examples of such parameters include the residual energy and distance (which also requires location-awareness). Naturally, the energy saving approach in these protocols consists of balancing the node usage. Keeping this balance comes with a cost, given that overuse of the intermediate nodes is not prevented and maintaining the inter cluster communication ends up increasing the energy consumption in the nodes. Other WSN Routing Protocols [16]–[18] follow a different approach by using mobile nodes that move through constrained paths to collect information. ZEAL, the protocol in this paper also make use of this concept to collect information. Different to similar protocols, ZEAL do not require clock synchronization and provides an energy saving solution used in the data collection process.

3. The ZEAL Routing Protocol

In general, routing protocols are categorized as reactive or proactive, in relationship to the way they create their routes to a destination. In proactive protocols, the nodes create routes in advance, while in reactive protocols the nodes create routes only when they are requested to send data. These characteristics make proactive protocols best suited for non-mobile scenarios while reactive protocols are used in scenarios with high mobility. The protocol discussed in this paper (ZEAL) is a hybrid, proactive in nature but designed to be used in scenarios with one or more mobile nodes that collect data from the network. According to their behavior, nodes in ZEAL can be classified into 3 types (Fig. 2):

- *Sink nodes.* Mobile nodes in charge of diving the space into routing zones and collect data from these zones through the subsink nodes.
- *Subsink nodes.* Nodes that are at some point within direct communication area of a sink. Only subsinks can transmit data to a mobile sink directly.
- *Member nodes.* Nodes that are not within a direct communication area of a sink node and require additional node relays to reach a subsink. There are one or more intermediate nodes along the path from a member node to a subsink.

In ZEAL one or more mobile sinks move at constant speed through a constrained path. The path-constrained mobile sinks do not necessarily have to move back and forth between a start point and an endpoint. Mobile sinks are allowed to move unidirectionally, but are required to move along a constrained path multiple times. For the following discussions, we assume for simplicity that mobile sinks move at constant speed back and forth between a start point and an endpoint (cyclical movements). On the first cycle, ZEAL forms independent *routing zones*, and in subsequent cycles, it collects data from these routing zones. Although the constant speed requirement can be seen as a constraint of the protocol, it can also bring some benefits. ZEAL uses the constant speed of the sink to create semi-equal routing zones and optimize the data collection process and energy saving. In our evaluations, member nodes and subsinks do not have any particular characteristics. Neither GPS module nor renewable external energy sources are assumed. On the other hand, sink(s) are assumed to be arbitrary powerful nodes according to the application requirements and deployment characteristics. For example, a sink installed on a circular conveyor belt that allows the sink to achieve the desire cyclical movements at constant speed. Another option is the use of sinks on autonomous vehicles [19] capable of moving in cycles over large areas. Mobile sinks can also make use of ex-

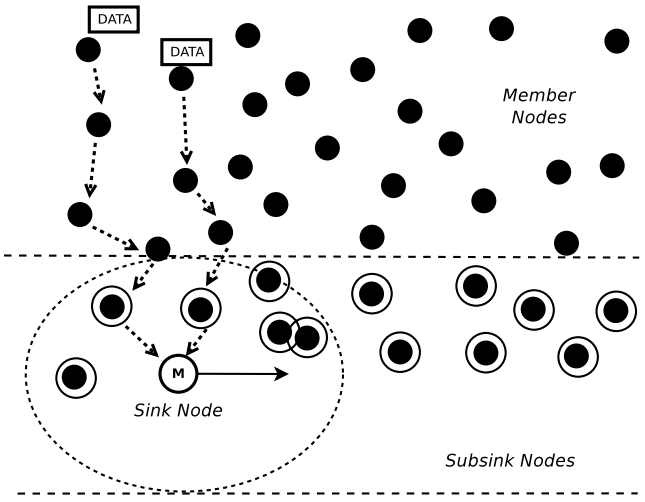


Fig. 2 Type of nodes used in ZEAL. A sink node moves exchanging information with the subsink nodes, while the member nodes use a multi-hop communication to reach the subsink nodes.

isting infrastructure. For instance, a mobile sink node can be placed in an aerial tramway (ropeway car). Aerial tramways move naturally in cycles with constant speeds. They could collect information from sensor nodes scattered around a mountain. For example, the information could describe the values in humidity from different levels of the mountain, or even describe the amount of snow in different areas on the mountain and be used as an early detection of avalanches in winter.

The ZEAL protocol requires two phases to function: The *setup phase* and the *data collection phase*.

3.1 Setup Phase

The objective of the setup phase is to create independent routing zones, populate the routing tables and select the subsink in each zone. The number of zones assigned to each sink is determined by solving an optimization problem. Let S and Z be the desired total number of sinks and zones selected by the sensor network administrator. The number of zones z_i for each mobile sink i is assigned to satisfy the following conditions ($i = 1 \ldots S$):

$$min\{max(|z_i - z_j| : i \neq j, i, j = 1 \ldots S)\}$$
$$subject\ to\ z_i - z_j \geq 0 \qquad i \geq j, i, j = 1 \ldots S$$
$$\sum_{i=1}^{S} z_i = Z$$

The objective is to have the smallest difference in the number of zones assigned to each mobile sink. Each mobile sink has an identifier attached to it. The first constrain ensures that the sink with the smallest identifier has the highest priority. That is, it will be assigned with a larger number of zones if we cannot divide the number of zones equally.

According to the movement of the mobile sinks the following duties are accomplished:

First half of cycle 1. During the first half of cycle 1, mobile sinks use *BCST1* messages similar to a beacon mechanism. All mobile sinks move with constant speed while broadcasting *BCST1* messages containing the assigned zone ID. The zone ID to be broadcasted is determined by dividing the total time of half cycle by the predefined number of zones assigned to that mobile sink. For example, in a scenario where half cycle equals to 90 seconds and the total number of zones are 3, the mobile sink broadcasts the zone ID 1 the first 30 seconds, zone ID 2 for the next 30 seconds and zone ID 3 for the last 30 seconds. Every node receiving a *BCST1* message becomes a subsink candidate and responds to the originator mobile sink with a *UCST1* message. When a mobile sink receives a *UCST1* message from the subsink candidate for the first time, the reception time is saved in the mobile sink as its *time range start*. The reception times of consecutive *UCST1* messages coming from the same node are saved as *time range end*. In this way, by the end of the first half of cycle 1, the mobile sink becomes aware of the start and end of the *communication range* of each subsink candidate. In a dense populated topology, multiple candidate subsinks are located close to each other and therefore, there is a high probability their communication ranges overlap with each other. In order to communicate effectively, a mobile sink should only receive or transmit information with one subsink at any given time. If every subsink in range transmits data simultaneously, packet loss caused by collisions are unavoidable. To minimize this problem, the mobile sinks use the captured communication range times to allocate specific *time slots* in which each subsink can communicate exclusively with a mobile sink. A subsink time slot has a start and end within the boundaries of the communication ranges previously captured. Both communication range times and time slots assignment are saved in their corresponding mobile sink(s). The time slot assignment based on the communication range times of candidate subsinks takes place exactly at the end of the first half of cycle 1. Time slots can be assigned in multiple ways [16], the following algorithms can be used in ZEAL:

- *Minimum Time Assignment (MinTA)* gives priority for the overlapping transmission times to the last subsink in the overlapped area. This algorithm is a simple approach and the resulting time slots are greatly unbalanced.
- *Shared Time Assignment (ShareTA)* attempt to divide the overlapped range into even time slots for all the subsinks located in the communication range of the mobile sink. However, the resulting assignment can still contain overlapped time slots in cases where subsinks are too close together or over each other (Fig. 3-(B)). In addition, some subsinks can be assigned with a time slot too small to be useful.
- *Selective Time Assignment (SelectiveTA)* This new algorithm is specifically designed for ZEAL. It takes advantage of the subsink candidates preselection used

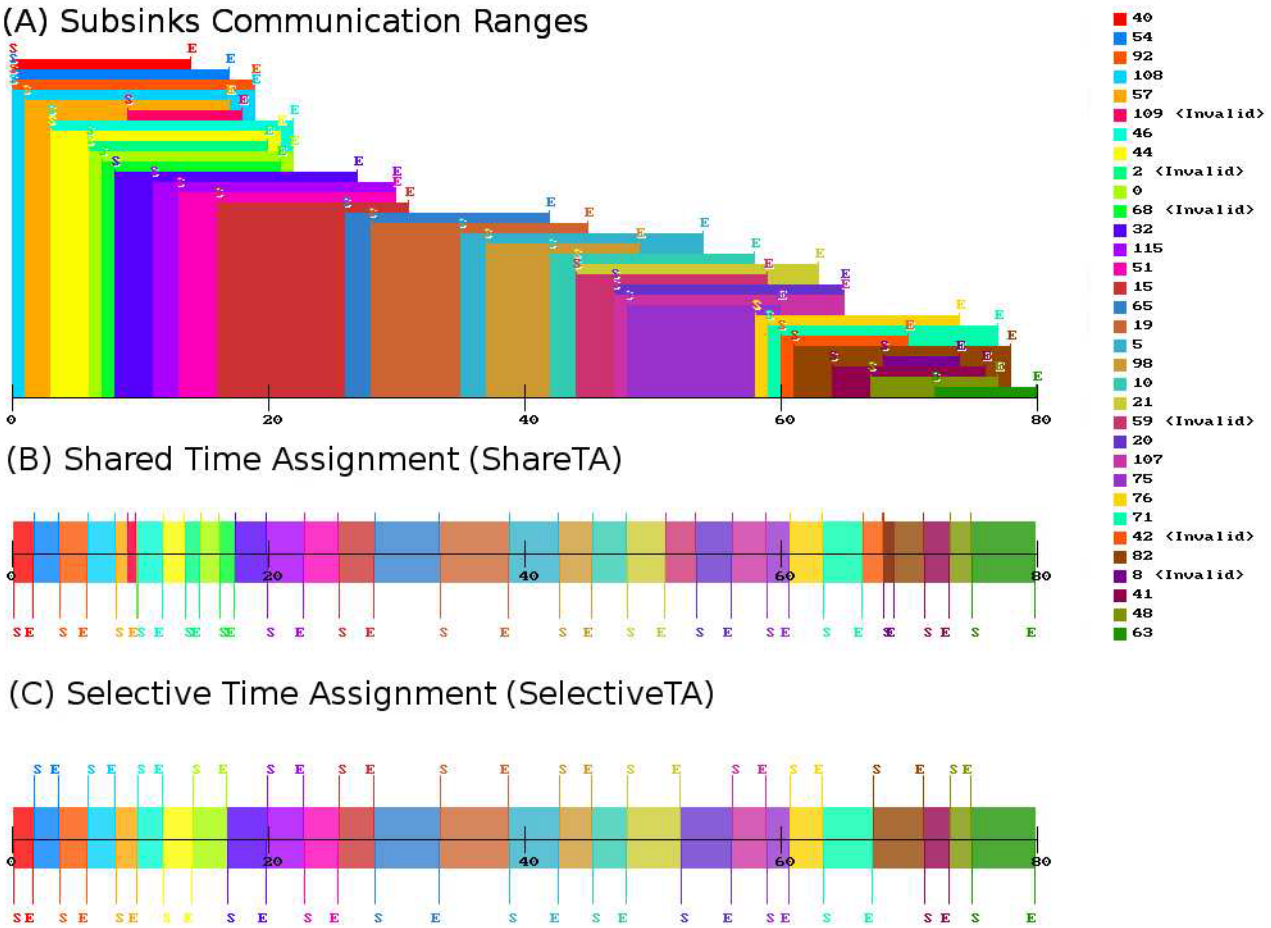


Fig. 3 Subsink communication ranges and time slots assignment algorithms.

in ZEAL. Different to algorithm such as ShareTA, it achieves a real exclusive communication time slot assignment to each subsink. SelectiveTA use 2 filters:

Filter 1. After a preliminary assignment of the time slots, selectiveTA removes any remaining overlapped time slots. When multiple candidate subsinks with assigned slots overlap, the candidate subsink with the largest assigned time slot takes priority.

Filter2. An x percent of the candidate subsinks with the smallest time slots are removed in each zone. The variable x is a parameter and set by the administrator. Not all the zones posses the same number of candidate subsinks which means that only the zones that have enough candidate subsinks will be affected by this filter. For example, when a zone with only 2 subsinks and $x = 20\%$, the number of subsinks is not affected by this filter. A zone with 5 candidate subsinks and $x = 20\%$ would be reduced by 1 subsink. By using a small value of x, zones with a large number of candidate are reduced while zones with a small number of candidates remain unaffected. The time slots of the removed candidate subsinks are distributed among the contiguous subsink candidates. Removing candidate subsinks increases the length of the time slots of the remaining candidate subsinks which are closer to the mobile sink. Furthermore, there is a positive side effect that comes from filtering less effective candidate subsinks. No routes are created to removed candidate subsinks. Network traffic caused by control packets is reduced and energy is preserved as a result of not creating these routes. A real example of the subsink time slot assignment for ShareTA and SelectiveTA is shown in Fig. 3. Figure 3-(A) shows the subsink communication ranges as captured by the mobile sink. The horizontal axis represents time. The mobile sink begins to move at time 0 and reaches the end of its movement path at time 80. "S" and "E" denote the start time and end time of the communication ranges between each node and the mobile sink, respectively. For example, subsink 40 communication range is between time 0 and 15 seconds (red color). Fig. 3-(B) shows the slot time assignment after executing ShareTA. It is also possible to observe the resulting overlapped time slots in some cases (subsinks 109 and 82). In Fig. 3-(C), SelectiveTA time slots assignment can be seen. Compared to Fig. 3-(B), SelectiveTA filter 1 removed the overlapped time slots. Additionally, as previously described, filter 2 eliminated and redistributed the time slots of some subsinks. Summarizing, SelectiveTA ef-

fectively reduces the number of candidate subsinks in the zone by taking away the least suitable candidate subsinks. All subsink time slots filtered by SelectiveTA are label "Invalid" next to the subsink number.

Second half of cycle 1. In this half of cycle 1 the time slots resulting from the algorithms in the first half of cycle 1 are used. From this point, all candidate subsinks are now considered subsinks. Immediately after obtaining the subsink time slots, a calculation of the member requirements ($Mreq$) take place. $Mreq$ represents the number of member nodes from that a subsink can ideally collect data. This number depends on the size of its assigned time slot. The $Mreq$ calculations take place inside the mobile sink using Eq. (1) [16], [17].

$$Mreq = \frac{dt * at}{ds * mt} - 1 \tag{1}$$

The variable dt represent the data rate at which subsinks transmit to the mobile sinks. The variable at is the assigned time slot obtained after the execution of the time assignment algorithms. The variable ds is the application data rate used to transmit to the subsinks. Finally, the variable mt is the total trajectory time of the mobile sink. After the Mreq of each subsink is calculated, the mobile sink heads back to its original position while sending *UCST2* messages to each one of the subsinks. By sorting the assigned time slots by start or end time it is possible to send *UCST2* messages to each one of the subsinks sequentially. The sorting criteria (ascending index or descending index) changes according to the moving direction of the mobile sink. We call this process the *index mechanism*. When a subsink receives its *UCST2* message, it triggers the sending of the subsink's first *BCST2* message containing the subsink zone ID, hop count and its Mreq number. Any member node receiving the *BCST2* message, processes it and if some conditions are fulfilled, re-broadcasts the *BCST2* message (Algorithm 1). The *BCST2* messages create the routing zones, distribute the subsinks Mreq information and by using the hop information, create the routes of each member node to the subsinks in its zone. Figure 4 shows an example of the resulting zones after the setup phase is completed (end of cycle 1). Before transmitting data packets in the *data collection phase* each member node must select the subsink to transmit data.

$$Pr = \alpha * Mreq + \frac{1-\alpha}{hopCount} \tag{2}$$

Member nodes assign a priority Pr to all subsinks in its zone using Eq. (2). The variable α is a weight value between 0 and 1. The subsink with the highest priority is selected as the subsink destination.

3.2 Data Collection Phase

After cycle 1 every node except for the mobile sinks, generates data at the beginning of every cycle. Prior to the data generation, the routing tables in all nodes are purged of entries to destinations of nodes belonging to different zones to

Algorithm 1: RecvBcst2 algorithm for dividing the network in zones, creating the shortest path from each node to their subsinks and distributing mreq.

```
1  function RecvBcst2 (msgHeader);
   Input : A packet with a msgHeadear containing Bcst2
           parameters
2  if sinkFlag then
3  |  return;
4  end
5  if localAddress == subsink then
6  |  return;
7  end
8  zoneId = msgHeader.getZoneId;
9  mreq = msgHeader.getMreq;
10 hop = msgHeader.getHopCount;
11 subSink = msgHeader.GetSubsink;
12 new rangeEntry(zoneId, mreq, subsink);
13 if rangeEntry not in RangeTable then
14 |  Add entry to RangeTable;
15 end
16 if RoutingTable == empty then
17 |  localZone = zoneId;
18 |  minHop = hop+1;
19 |  newEntry(zoneId, hop+1, subsink);
20 |  Add newEntry to RoutingTable; SendBcst2 (subsink,
   |    hop+1, zoneId);
21 else if localZone == zoneId then
22 |  Search entry with zone == zoneId and destination ==
   |    subsink;
23 |  if entry not found then
24 |  |  newEntry(zoneId, hop+1, subsink);
25 |  |  Add newEntry to RoutingTable;
26 |  |  SendBcst2 (subsink, hop + 1, zoneId);
27 |  else if entry.getHopCount ≥ hop+1 then
28 |  |  updateRoutingTableEntry(hop+1, sender);
29 |  |  SendBcst2 (subsink, hop+1, zoneId);
30 |  end
31 |  if minHop ≥ hop+1 then
32 |  |  minHop = hop+1;
33 |  end
34 else
35 |  if minHop ≥ hop+1 then
36 |  |  localZone = zoneId;
37 |  |  minHop = hop+1;
38 |  |  updateRangeTable(subsink, zoneId);
39 |  |  newEntry(zoneId, hop+1, subsink);
40 |  |  Add newEntry to RoutingTable;
41 |  |  SendBcst2 (subsink, hop+1, zoneId);
42 |  |  while end of RoutingTable do
43 |  |  |  if RoutingTableEntry.zone == zoneId
   |  |  |    and flag == invalid then
44 |  |  |  |  RoutingTableEntry.setFlag = valid;
45 |  |  |  |  SendBcst2 (subsink, hop+1, zoneId);
46 |  |  |  end
47 |  |  end
   |  |  /* Inform neighbor nodes that the current
   |  |     node is not longer available for the
   |  |     old zone                                  */
48 |  |  BcstZoneChangeAck(localZone);
49 |  else
50 |  |  newEntry(zoneId, hop+1, subsink);
51 |  |  Add newEntry to RoutingTable with invalid flag;
52 |  end
53 end
```

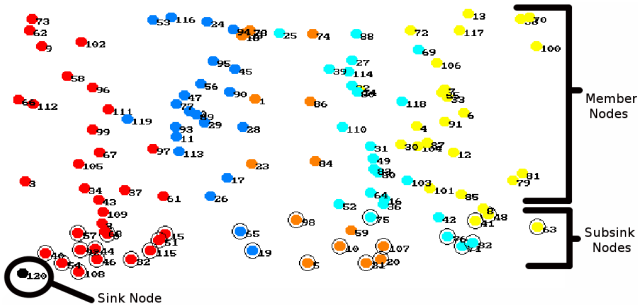


Fig. 4 An example of zone partitioning. The network is divided into 5 zones after executing the setup phase with a single mobile sink at bottom. Double circles denote the nodes selected as subsinks. The rest of the nodes are considered member nodes.

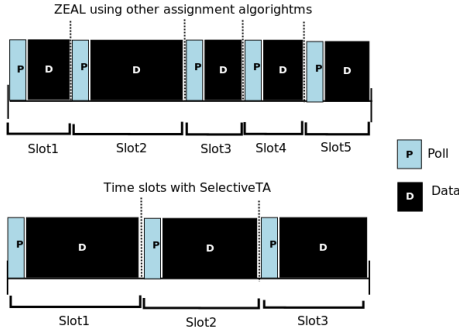


Fig. 5 Impact of SelectiveTA in the data collection phase using the poll mechanism.

their own. Immediately after, the data is sent to their designated subsink (obtained based on the Mreq). When the data reaches a subsink it is queued until requested by a mobile sink. In the *data collection phase* the mobile sinks continue their cyclical movements along their constrained paths while collecting data generated by the member nodes queued at the subsinks. To accomplish this, the mobile sinks send *POLL1* messages to each one of the subsinks on their trajectory by using the index mechanism. The approach is simple in design. The mobile sink sends a *POLL1* message using the index mechanism previously described. A *POLL1* message contains the size of the time slot assigned to that specific subsink. The subsink which receives the *POLL1* transmits *DATA* packets from its queue to the mobile sink until the queue is depleted or the assigned time slot runs out.

Figure 5 shows the *poll mechanism* described above. Additionally to the benefits of the SelectiveTA algorithm described in the setup phase, the filters used by SelectiveTA have an impact on the data collection process. A *POLL* message takes a portion of the time slot assigned to a subsink, the greater the number of subsinks the less time is used to actually transmit data. Reducing the overlap and the number of subsinks effectively increases the available data transmission time between the subsinks and the mobile sinks.

3.2.1 Duty Cycling

Node idling is arguably one of the most energy consuming

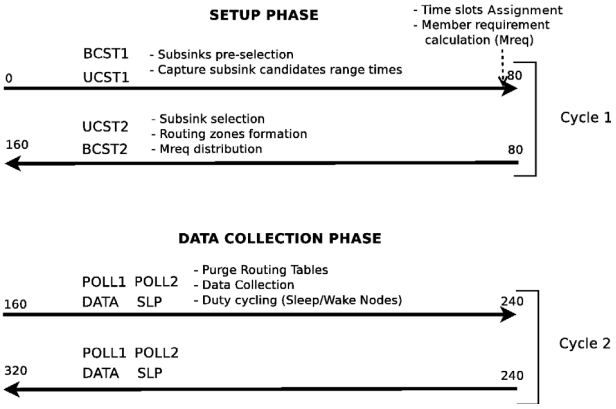


Fig. 6 ZEAL Phases with 160 seconds cycle.

functions in a WSN [20]. ZEAL introuduces an energy saving mechanism to the data collection phase. While the poll mechanism is mainly done to request *DATA*, it is also used to facilitate a "sleep mode per zone". In the data collection phase, at any given moment the mobile sink is in communication with one subsink in a single zone. Therefore, it is possible to put to sleep the nodes in zones whose subsinks are not actively transmitting *DATA* to the mobile sink. Setting a zone in sleep mode is done by having the mobile sink send a *POLL2* message instead of a *POLL1* when in range of the last subsink in a zone. A *POLL2* message contains the *sleeping time* and the *time slot time*. The sleeping time is equal to the time the mobile sink takes to return to the same position (*(half cycle time - current time in cycle)* 2*).

The subsink receiving this message will decrease the sleep time by one second and broadcast a *BCSTSLP* message containing the zone id and the sleeping time. After completing these steps, the subsink sleeps for the amount of time specified by the sleep time. The rest of the nodes receiving the *SLP* message will rebroadcast the message and enter into a sleep mode in a similar way. The process repeats until the zone is completely in a sleep mode. Figure 6 shows an example of the ZEAL protocol phases. The arrows represent the movement of the mobile sinks in a 160 second cycle.

4. Evaluation

All experiments were made using the network simulator ns-3 (version 3.25). In all the experiments, the wireless nodes were random uniformly distributed on a 400m x 200m virtual rectangular area using 10 different random seeds which creates 10 topologies. The mobile sink moves through the bottom of the rectangular area (node distributions are similar to the one shown in Fig. 4). Results are averaged in cases where multiple seeds are used. The number of nodes in the network is ranging between 120 and 200. The number of zones is set to 5. According to our preliminary simulations with these configurations, average number of subsinks in each zone is about 5. Therefore, we use 20% as x, i.e. the worst 1 subsink candidate is averagely removed in each zone.

Table 1 Differences between MASP and ZEAL protocols.

Characteristics	ZEAL	MASP
Minimum number of rounds	1.5	3
Automatic zone distribution	yes	no
Energy saving function	yes	no
Nodes clock Syncronization Required	no	yes
Data collection mechanism	Poll mechanism	Synchronized transmission
Time assignment algorithms	SelectiveTa ShareTa MinTa	ShareTa MinTa
Inefective subsinks filter	no	yes
Independent zones per sink	yes	no
Precise time assignment times	no	yes
Constrained path sink movement	yes	yes

The Maximum Amount Shortest Path (MASP) protocol [16] is chosen as a target of comparison in our simulations. MASP, similar to ZEAL is a multi-hop routing protocol that uses one or more mobile sinks moving at constant speed. Both, MASP and ZEAL, divide the the network into zones and have member nodes with routes to each subsink in their zone. One key difference of MASP from ZEAL is to exploit the complete movement prediction of the mobile sink for subsinks to transmit data packets to it when it enters in range of the subsinks (synchronized transmissions between subsinks and mobile sinks). This approach achieves more accurate transmission time slots than ZEAL's poll mechanism, but it assumes perfect synchronized clocks among all nodes in the network. This is a critical constraint of MASP because it makes a real implementation less feasible and, with a higher level of complexity (additional synchronization methods are required). Key differences between MASP and ZEAL are summarized in Table 1.

Several WSN protocols have been proposed to reduce node energy consumption [12]–[15]. These protocols employ a cross-layer design, and therefore, combine the Mac layer functions with the network layer functions to save energy consumed by nodes. On the other hand, ZEAL is a network layer protocol. All mechanisms of ZEAL, e.g. zone division, duty cycling and so on, are operated in the network layer. ZEAL can take advantage of the hierarchical, modular nature of network protocol design. For example, it is possible to adapt ZEAL in a future implementation to use LR-WPAN. ZEAL can use its duty cycling mechanism as well as LR-WPAN duty cycling mechanism (*contikiMac*) for further energy saving. It is difficult for existing cross-layer WSN protocols to use another layered protocol together because cross-layer protocols and layered protocols conflict with each other. Design approaches of both ZEAL and existing energy-efficient WSN protocols are basically different. Therefore, existing energy-efficient WSN protocols cannot be applicable to the assumed environment in this paper as an alternative to ZEAL.

In our evaluations wireless radios are set to a maximum range of 52m using a constant propagation delay. In all our experiments the packet size of the application generated data is set to 1024 bytes; however, both MASP and ZEAL have an additional overhead of 5 bytes to redirect the packets to the appropriate subsinks. Application data is only generated for 1 second at the beginning of every cycle in the data collection phase. All nodes have an initial energy of 3000 J (equivalent to a single AAA NiMH battery). Energy evaluations used the default energy model in ns-3 [20]. This model consists of energy sources that provide the energy (a basic linear ideal energy source model) and the devices that consume the energy from these sources (in this case, the Wi-Fi radio energy model). We used the default energy consumption values used by the energy source model: RxCurrent = 0.313 Amp, TxCcurrent= 0.38 Amp, IdleCurrent = 0.273 Amp, CcaBusyCurrent = 0.273 Amp. Ideally, LR-WPAN (IEEE 802.15.4) is used in WSN evaluations. Nevertheless, at the time of the evaluation the LR-WPAN ns-3 module was still undergoing development and there was no official WSN protocols that could be tested against ZEAL in the repositories. Additionally, the equivalent LR-WPAN energy model has yet to be implemented in ns-3. For this reason, in order to evaluate ZEAL's delivery ratio and energy, we made use of Wi-Fi (IEEE 802.11b). In all the experiments the mobile sinks moves at constant speed of 5 m/s (unless indicated otherwise).

Figure 7 shows average delivery rate performance with different number of nodes. In the experiment, the results are an average of the 10 random uniformly distributed topologies. The number of nodes are increased by adding additional nodes to the topology. For example, the topology of 125 nodes is created by adding 5 additional nodes to the topology of 120 nodes in the same way that the topology of 130 nodes includes all the nodes of the topology of 125 nodes. Additional nodes create additional routes as well as the creation of different zone shapes in each protocol. For this reason, the delivery ratio fluctuates while the number of nodes increases as shown in Fig. 7. It is also possible to observe that MASP has a higher delivery ratio than ZEAL in some number of nodes, though ZEAL achieves a higher delivery ratio, comprehensively.

One example of this can be observed in Fig. 8. In this experiment, a single seed (1 topology) and a single data delivery is used to demonstrate ZEAL's higher delivery ratio. Additionally, in all the experiments, ZEAL is actually sending more packets, and therefore, has a higher chance of packet delivery failure in the experiment. ZEAL uses only 1 cycle to set up the division of the network and the creation of the routes while MASP require at least 2 cycles to accomplish the same setup. Consequently, ZEAL is producing

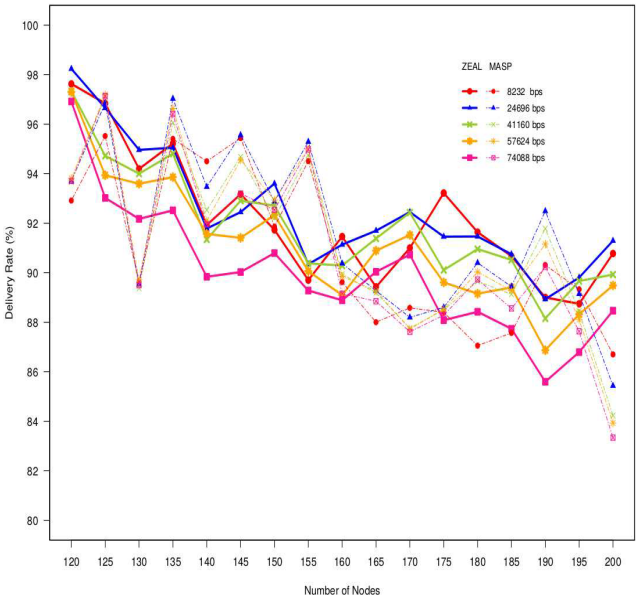


Fig. 7 Average data delivery rate using an increasing application data generation rate and number of nodes. Total duration: 3 cycles. Using 10 uniform random distributed topologies.

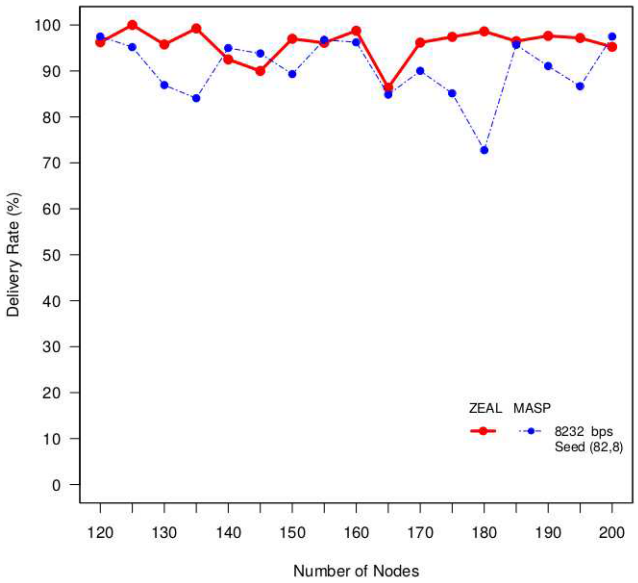


Fig. 8 Data delivery rate using a single seed (1 topology). Total duration: 3 cycles.

packets for at least 1 more cycle than MASP. In both protocols, the delivery rate decreases as the number of nodes increases. This is because the number of simultaneously transmitted packets increase the chances of collisions that lead to route failure during the setup phase. However, ZEAL transmits less simultaneous packets in the setup phase than MASP because ZEAL creates zones by using *UCST2* messages separated by the assigned slot times (Fig. 6) instead of constant broadcasts used by MASP. As a result, route failure is more likely to occur in MASP than in ZEAL. This is the main reason why ZEAL achieves a higher delivery ratio than

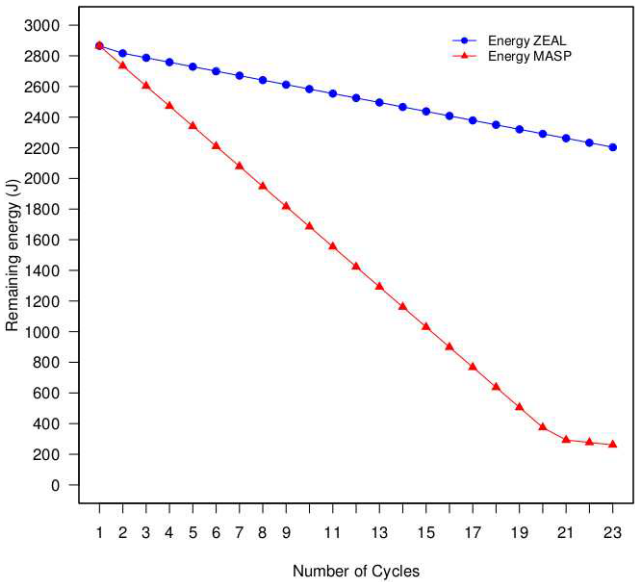


Fig. 9 ZEAL and MASP average remaining energy after 1 hour. Application data is generated at the beginning of every cycle.

MASP. Figure 9 shows the average remaining energy of the nodes in the network. In this experiment we used exclusively a data application generation rate of 8232 bps (equivalent to 1 packet/node in 1 cycle). We ran the experiment for 23 cycles which corresponds to 1 hour in real time. Since MASP lacks an energy saving feature, the energy consumed by idling nodes quickly decreases the lifetime of the network. On the other hand, ZEAL is able to save energy by putting to sleep any zone that is not in range of the mobile sink.

In ZEAL, subsinks deplete their energy earlier than member nodes. This is an unavoidable problem in WSNs using subsinks (sometimes called cluster heads). A rotating subsink mechanism could save some amount of energy but increases the complexity of the system as intra-subsink communication or synchronized scheduling mechanisms are required. In ZEAL, we opted for a simple implementation. Using the topology shown in Fig. 10-(A), it is possible to observe the variance of remaining energy among the member nodes in the same zone remains relatively close even after 24 cycles (Fig. 10-(B)), or after 72 cycles (Fig. 10-(C)). The same thing may be said of subsinks in the same zone. The subsink nodes have nearly the same amount of remaining energy. Zones are independent from each other and a single subsink in each zone provides service to some of the member nodes in the zone. If a subsink depletes its energy and, as a result, an energy hole arises, only the member nodes served by it suffer the disruption. Not all the nodes in the zone are affected by the energy hole. Therefore, energy holes in ZEAL are not as critical as in other protocols (The whole network is not compromised). Since, subsinks energy levels are lower, but close to the remaining energy of the member nodes, they can be used as an early indicator of the energy levels in the zone. Regarding a maintenance procedure in ZEAL, periodical execution of the setup phase can help to

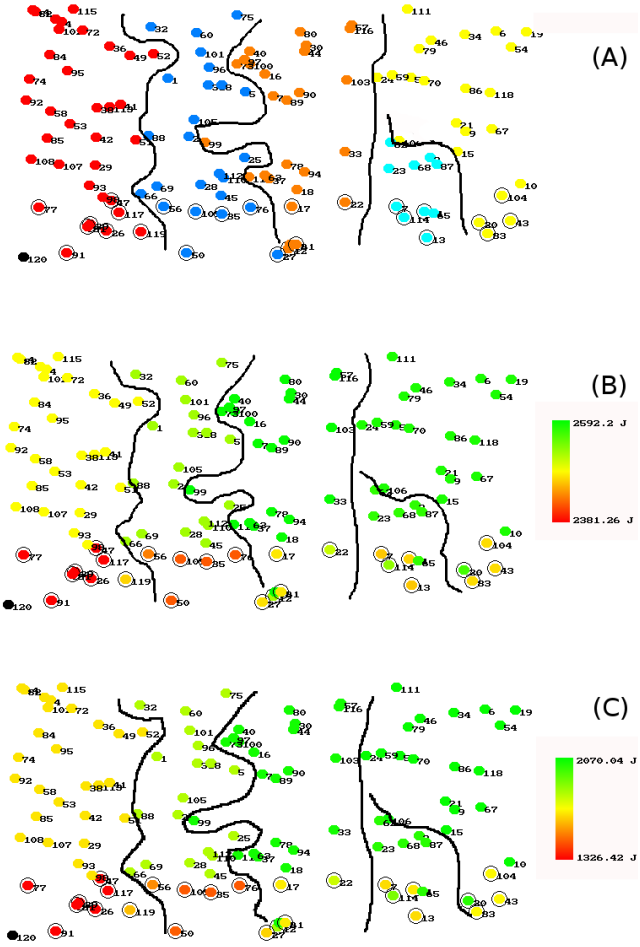


Fig. 10 (A) Division of nodes into 5 zones using seed(20,6) with ZEAL protocol. (B) Energy Remaining after 1 hour (24 cycles). (C) Energy Remaining after 3 hours (72 cycles). Energy colors are based on the maximum and minimum residual node energy in each test.

integrate new nodes to the network or discard unavailable ones.

Figure 11 shows the average remaining energy as well as the maximum and minimum values registered after 3 cycles. While ZEAL has more energy variance due to the duty cycling mechanism, the minimum energy consumption values are still noticeably lower than MASP. MASP average, minimum and maximum remaining energy values are close because the most significant source of energy consumption is the idle state of the nodes. The performance of ZEAL and MASP with only 3 cycles can be seen in Fig. 12. When few data packets are produced the number of delivered packets between ZEAL SelectiveTA and SharedTA is similar. However, as the number of packets increases, SelectiveTA has a higher delivery rate because it assigns more time to the best subsinks while discarding the ones with the worst time slots. In both SelectiveTA and ShareTA, ZEAL outperforms MASP because its setup phase is only 1 cycle. A small setup phase saves time and energy and can be particularly beneficial in cases where repeated execution of the setup phase is required (e.g. A topology that has periodic node changes).

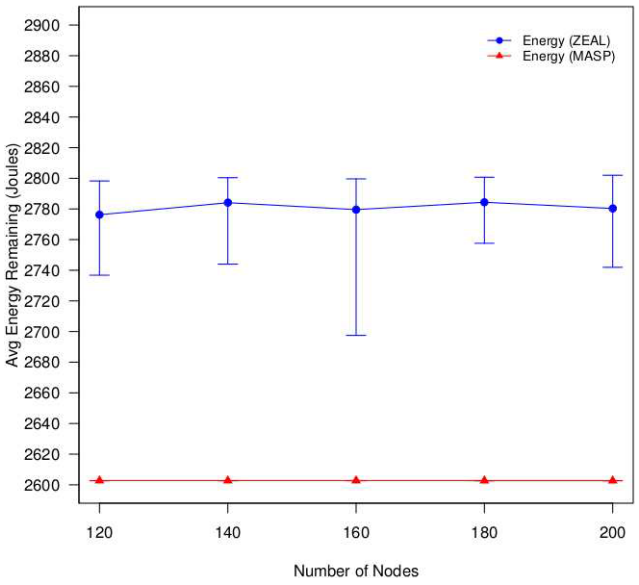


Fig. 11 Average remaining energy of 10 different seeds and an increasing number of nodes. Maximum and minimum values caused by the duty cycling can also be seen in ZEAL. Total duration: 3 cycles.

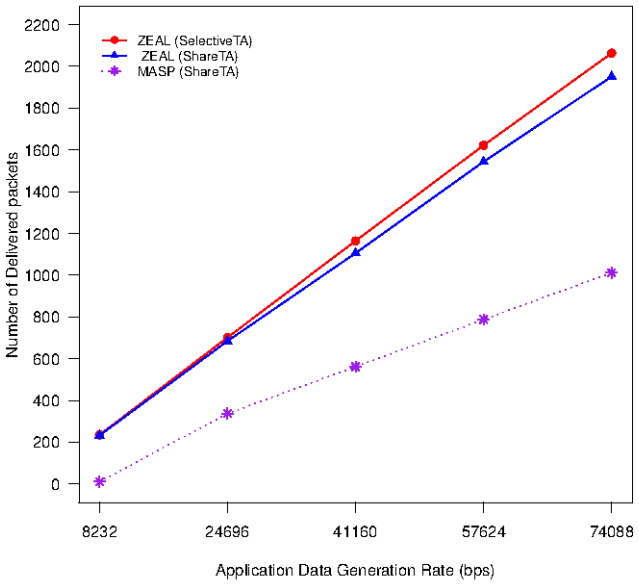


Fig. 12 Delivered packets with different time slots assignment algorithms.

In Fig. 13, data generation rates of 8232 bps and 74088 bps are used to test the differences between the time slot assignment algorithms in ZEAL and MASP. As mentioned previously in Fig. 12, with a higher number of packets (in this case caused by a higher number of nodes and data generation rate) the differences between SharedTA and SelectiveTA become more obvious. These differences are specially important when there is a high number of nodes close to the mobile sink even if the data generation rate is low. In the case of SharedTA, considering every node in the path of the mobile sink as a subsink translates into an

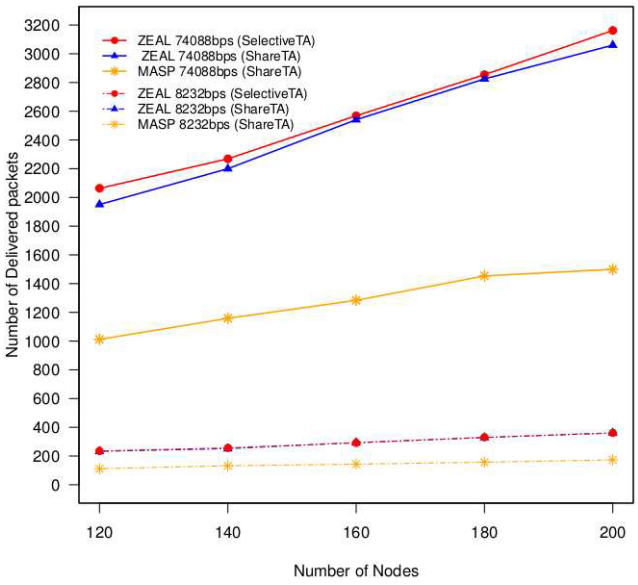


Fig. 13 Time slots assignment with different number of nodes. ZEAL with SelectiveTA performs better as the number of packet increases.

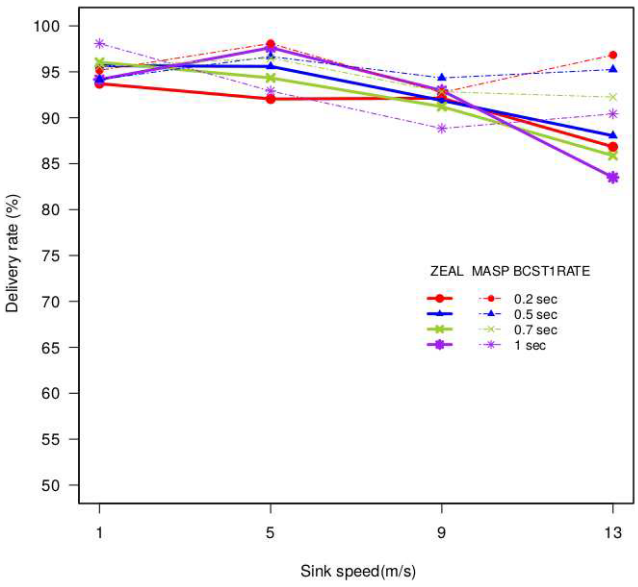


Fig. 14 Data delivery rate using a single sink at different speeds. Execution time 3 cycles.

excessive number of small slots with a significant chance of overlap (similar to the ones shown in Fig. 3).

Figure 14 shows the results of our experiments using a single sink moving at the bottom of a topology formed by 120 nodes. We tried different broadcast 1 message (*BCST1*) rates and sink speeds to conduct our experiments. As explained in Sect. 3.1, *BCST1* are used as a beacon mechanism to determinate subsinks communication ranges in respect to a sink and their capabilities for the job as subsinks. The smaller the *BCST1* rate (interval) the more *BCST1* messages are generated in a single second. This generates more precise subsink communication time slots, but also increases the number of

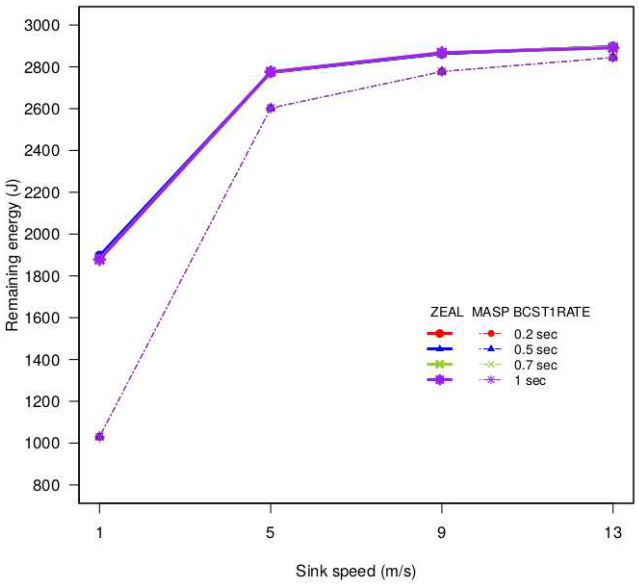


Fig. 15 Average remaining energy using a single sink at different speeds. Execution time 3 cycles.

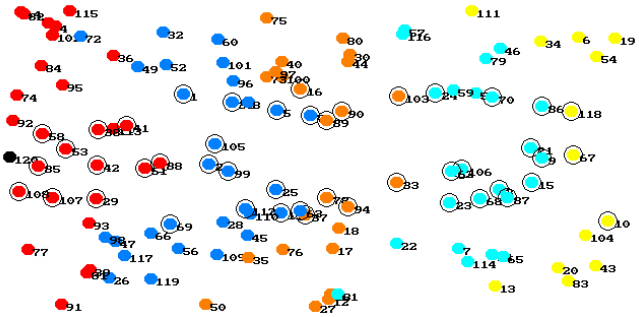


Fig. 16 ZEAL protocol in a topology of 120 nodes in a Uniform Random Distribution (seed(20,6)). The sink move along the middle of the topology.

broadcast 2 (*BCST2*) messages (specifically in the case of the MASP protocol). To generate more precise time slots with a minimum generation of *BCST2* in MASP, the value of the *BCST1* rate should be inversely proportional to the value of the sink speed. The experiment shows that ZEAL is affected by the speed of the sink. A faster sink movement implies that there is less amount of time for the *BCSTSLP* message to propagate. Subsinks might not be able to wake up on time before the arrival of the sink in subsequent rounds if the sleep time window is too small. Energy consumption is not affected by different values of the *BCST1* rate, but it is affected by the sink speed (Fig. 15). A faster sink will make smaller cycles and therefore, smaller sleep times. In experiments with a small number of cycles, the differences in energy consumption between MASP and ZEAL is relatively close. Nonetheless, the energy saving done in ZEAL becomes more significant in experiments with a higher number of cycles as previously shown in Fig. 9.

Figure 16 shows the topology previously used in Fig. 10-(A) but this time with the mobile sink moving along the middle of the topology using the ZEAL protocol. Naturally,

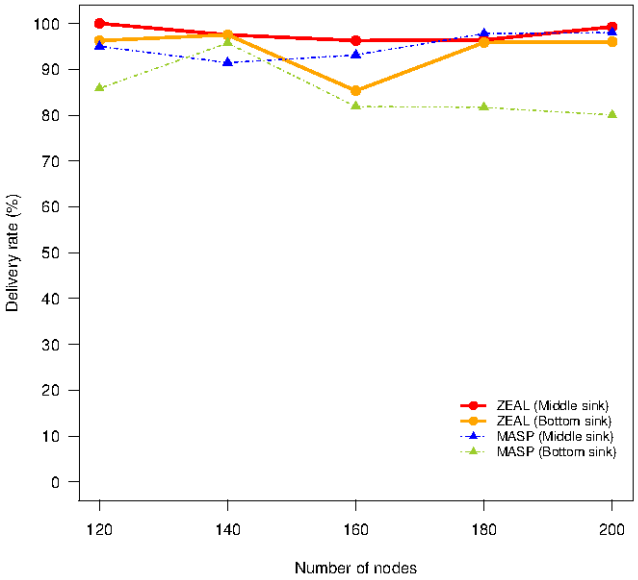


Fig. 17 ZEAL and MASP protocols data delivery rate with sink moving along the middle or bottom of the topology (seed(20,6)).

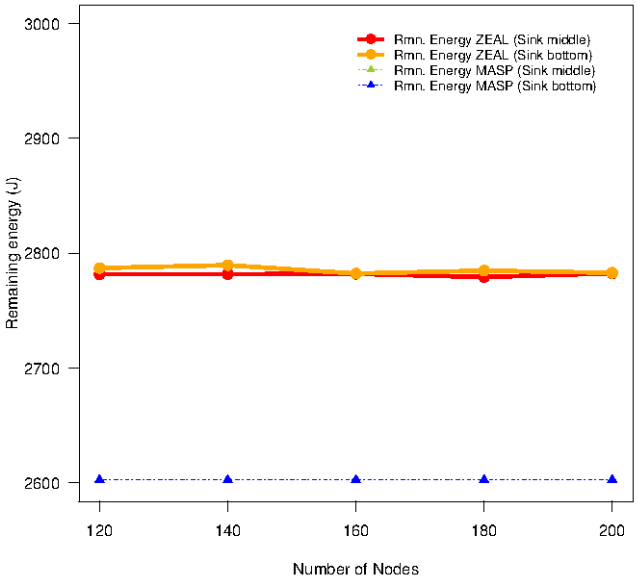


Fig. 18 ZEAL and MASP protocols remaining energy with sink moving along the middle or bottom of the topology. Seed(20,6), 5 zone.

the change creates a different zones distribution and subsinks locations. In Fig. 17, the resulting delivery rates using a sink passing through the middle or bottom of a topology (seed(20,6)) can be seen.

We can observe that the differences between MASP and ZEAL with the mobile sink moving at the middle are smaller than that with the mobile sink moving at the bottom. Although respective zones in both protocols using the mobile sink moving at the middle might have different shapes, a great deal of paths from member nodes to subsinks become 1 hop paths in both protocols. Consequently, the delivery rate of ZEAL is just slightly better than MASP. As for the remaining energy (Fig. 18) ZEAL is superior to MASP in both cases. The result of ZEAL with the mobile sink at the bottom is lightly better than ZEAL with the mobile sink at the middle, because the average amount of sleeping time of the bottom mobile sink is higher than that with the mobile sink moving at the middle. From the viewpoint of energy, the advantage of having the middle mobile sink is reducing the remaining energy variance between member nodes and subsinks in the same zone.

Figure 19 shows end-to-end delay performance of both ZEAL and MASP. This result is measured by using the *FlowMonitor* [21] ns-3 module. In this Figure, ZEAL can reduce end to end delay in comparison to MASP. ZEAL and MASP use mobile sinks to collect data, therefore, their end to end delay is affected mostly by the speed of the mobile sinks. Slow mobile sinks cause, in some cases, packets to be queued for long periods of time inside the subsinks (i.e. the packets collected in the zone from which the mobile sinks have already moved away). The end to end delay is affected not only by the time a mobile sink starts to collect the data from the subsinks, but also by the shape of the zones. Ideally, in the topologies where the mobile sink is located at the bottom, the

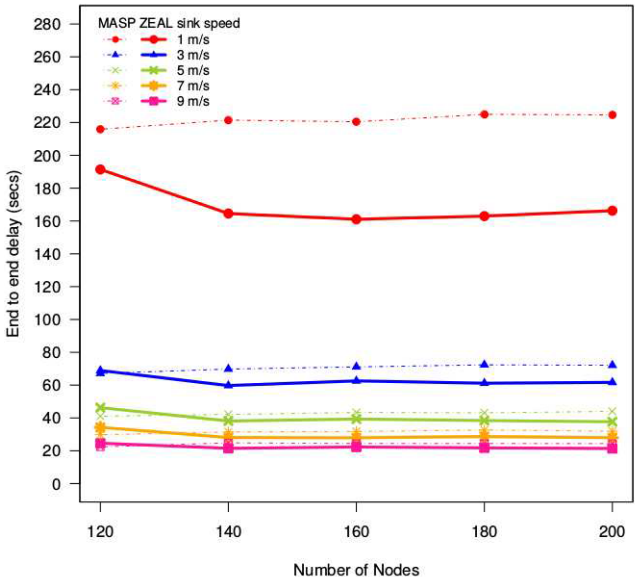


Fig. 19 MASP and ZEAL end to end delay using a single mobile sink moving at the bottom of the topology with different speeds. Seed(12,23), 5 zones, 3 cycles.

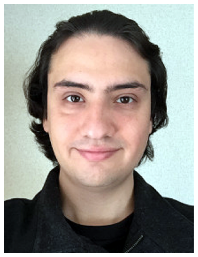
creation of the zones should be vertical and without nodes crossing other zones (a node from one zone should not be completely surrounded by nodes in other zone). Mobile sink speed has a great influence over the shapes of the zone in MASP because the slower mobile sinks moves, the greater the number of *BCST2* messages are generated and transmitted in the zones. ZEAL uses a single message per subsink (*UCST2* message) to trigger BCST2 (zones creation). This single message per subsink in ZEAL contributes to create zones more optimally comparing with MASP. For this reason, the better delay performance of ZEAL can be observed.

5. Conclusions

In order to mitigate the heavy energy consumption of nodes close to the sinks, we have proposed the ZEAL protocol, which uses a new communication time slot assignment algorithm (SelectiveTA) and a duty cycling mechanism. SelectiveTA can prevent the creation of non optimal routes and the duty cycling mechanism can reduce unnecessary idle time by introducing the sleep-wake cycle. In contrast to other constrained path routing protocols [16], [17], ZEAL is able to establish routes and a minimum setup phase with 50% fewer cycles. We conducted simulation experiments to analyze the effectiveness of our proposed communication time slot assignment algorithm and a duty cycling mechanism by comparing the performance of ZEAL with that of existing MASP protocol. Our simulations show that ZEAL improves upon MASP in both energy consumption and packet delivery rate. ZEAL is designed to work asynchronously so it does not use any additional location aware modules or synchronization algorithms that might be energy inefficient. The source code and graphical tools used in the development of ZEAL and MASP are available at [22]–[24].

References

[1] FAO, "How to feed the world in 2050," Viale delle Terme di Caracalla 00153 Rome, Italy, Food and Agriculture Organization of the United Nations, 10 2009.
[2] H.D. Leather and F. Phillips, The World Food Problem: Towards Ending Undernutrition in the Third World, 4th ed., Lynne Rienner Publishers, 2009.
[3] Boston food forest coalition, "Food forests," http://bostonfoodforest.org/, accessed June 21. 2017.
[4] Brooklyn Grange Farm, "RoofTop farms," http://www.brooklyngrangefarm.com, accessed March 3. 2017.
[5] City of Vancouver, "Vancouver urban farming guidelines," http://vancouver.ca/files/cov/urban-agriculture-guidelines.pdf, accessed May 27. 2017.
[6] L. Nguyen and S. Kodagoda, "Soil organic matter estimation in precision agriculture using wireless sensor networks," 2016 14th International Conference on Control, Automation, Robotics and Vision (ICARCV), pp.1–6, Nov. 2016.
[7] S. Prasad, S. Jaiswal, N.S.V. Shet, and P. Sarwesh, "Energy aware routing protocol for resource constrained wireless sensor networks," Proc. International Conference on Informatics and Analytics, ICIA-16, pp.121:1–121:7, ACM, New York, NY, USA, 2016.
[8] A. Chaudhary, "A cluster based wireless sensor network deployment for precision agriculture in dried and arid states of india," Proc. 2014 International Conference on Information and Communication Technology for Competitive Strategies, ICTCS'14, pp.60:1–60:3, ACM, New York, NY, USA, 2014.
[9] R.L. Gruner, D. Orazi, and D. Power, "Global versus local: An exploration on how vertical farms can lead the way to more sustainable supply chains," IEEE Eng. Manag. Rev., vol.41, no.2, pp.23–29, Second 2013.
[10] Nsam, "Ns-3 discrete event simulator," http://www.nsnam.org, accessed Jan. 07. 2017.
[11] J. Ivey, G. Riley, B. Swenson, and M. Loper, "Designing and enabling simulation of real-world GPU network applications with ns-3 and DCE," 2016 IEEE 24th International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS), pp.445–450, Sept. 2016.
[12] W.R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," Proc. 33rd Annual Hawaii International Conference on System Sciences, vol.2, p.10, Jan. 2000.
[13] A.A. Khan, N. Javaid, U. Qasim, Z. Lu, and Z.A. Khan, "Hsep: Heterogeneity-aware hierarchical stable election protocol for wsns," 2012 Seventh International Conference on Broadband, Wireless Computing, Communication and Applications, pp.373–378, Nov 2012.
[14] O. Younis and S. Fahmy, "Heed: A hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks," IEEE Trans. Mobile Comput., vol.3, no.4, pp.366–379, Oct. 2004.
[15] N. Javaid, S.N. Mohammad, K. Latif, U. Qasim, Z.A. Khan, and M.A. Khan, "Heer: Hybrid energy efficient reactive protocol for wireless sensor networks," 2013 Saudi International Electronics, Communications and Photonics Conference, pp.1–4, April 2013.
[16] S. Gao, H. Zhang, and S.K. Das, "Efficient data collection in wireless sensor networks with path-constrained mobile sinks," IEEE Trans. Mobile Comput., vol.10, no.4, pp.592–608, April 2011.
[17] A. Gallegos, T. Noguchi, T. Izumi, and Y. Nakatani, "Simulation study of maximum amount shortest path routing in wireless sensor networks using ns-3," 2016 Eighth International Conference on Ubiquitous and Future Networks (ICUFN), pp.198–204, July 2016.
[18] S.D. Dhamdhere, S.K. Guru, and C.D. Pune, "Robust data collection in wireless sensor networks with mobile sinks," IJCSIT, vol.5, no.4, pp.4999–5002, 2014.
[19] SMP robotics, "Security robots," http://smprobotics.com/, accessed May 24. 2017.
[20] H. Wu, S. Nabar, and R. Poovendran, "An energy framework for the network simulator 3 (ns-3)," Proc. 4th International ICST Conference on Simulation Tools and Techniques, SIMUTools'11, ICST, pp.222–230, Brussels, Belgium, Belgium, ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2011.
[21] G. Carneiro, P. Fortuna, and M. Ricardo, "Flowmonitor: A network monitoring framework for the network simulator 3 (ns-3)," Proc. Fourth International ICST Conference on Performance Evaluation Methodologies and Tools, VALUETOOLS'09, ICST, pp.1:1–1:10, Brussels, Belgium, Belgium, ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2009.
[22] A. Gallegos, "ZEAL protocol for ns-3," https://acidcube@bitbucket.org/acidcube/zeal, accessed March 3. 2017.
[23] A. Gallegos, "MASP protocol with syncronization for ns-3," https://acidcube@bitbucket.org/acidcube/maspsync, accessed March 3. 2017.
[24] A. Gallegos, "GraphNodes for ZEAL and MASP protocol," https://acidcube@bitbucket.org/acidcube/graphnodes, accessed March 3. 2017.



Alberto Gallegos received the M.S. degree from the graduate school of Information Science and Engineering, Ritsumeikan University, Japan in 2014. Since then, he has been pursuing his Ph.D. degree. His current research interest include but not limited to Wireless Sensor Networks, data mining, routing protocols and the internet of things.

Taku Noguchi received his B.E., M.E. and Ph.D. degrees in communications engineering from Osaka University in 2000, 2002 and 2004, respectively. He joined College of Information Science and Engineering at Ritsumeikan University in 2004, where he is currently an Associate Professor. His research interests include performance analysis and the design of computer networks and wireless networks. He is a member of IEEE, IEICE and IPSJ.

Tomoko Izumi received the M.S. and D.I. degrees in computer science from Osaka University in 2005 and 2008. She had worked at Nagoya Institute of Technology as a postdoctoral researcher in 2008. She has been an Assistant Professor of College of Information Science and Engineering, Ritsumeikan University since 2009. Her research interests include distributed algorithms and human interface. She is a member of IEEE, HIS, IEICE and IPSJ.

Yoshio Nakatani received his B.Sc. in sociology from Osaka University (Japan, 1981) and his Ph.D. (natural science) from Kobe University (Japan, 1989), respectively. In 1981–1998, he worked for Mitsubishi Electric Corporation as a researcher. In 1991–1992, he visited the Center for the Study of Language and Information (CSLI) of Stanford University (USA) as a visiting researcher. In 1998–2000, he temporarily transferred to the Dosys Corporation as a consultant engineer. In 2001–2004, he returned to Mitsubishi Electric Corporation and worked as a group manager of a customer consulting department of social information systems. In 2004, He moved to Ritsumeikan University (Japan) as Professor in the College of Information Science and Engineering. He was a vice dean of the college, in charge of international affairs, in 2005–2010, the department chairman of the Information and Communication Science in 2011, and the executive director of the Research Organization of Science and Technology of the University in 2012 and 2013. Since 2014, he has been the dean of his college. His research interests include application of Artificial Intelligence and Human-Computer Interaction to the fields of disaster mitigation information systems, intelligent transportation systems, and human Kansei (sensitivity) support systems. He is now President of the institute of Systems, Control and Information Engineers (Japan).