Wireless Netw (2012) 18:635–651
DOI 10.1007/s11276-012-0424-5

PSAR: power-source-aware routing in ZigBee networks

Metin Tekkalmaz · Ibrahim Korpeoglu

Published online: 3 March 2012

Abstract ZigBee is a recent wireless networking technology built on IEEE 802.15.4 standard and designed especially for low-data rate and low-duty cycle applications such as home and building automation and sensor networks. One of the primary goals of ZigBee is low power consumption and therefore long-living networks. Despite this goal, current network formation and routing protocols described in the ZigBee specification do not fully address power consumption issues. In this work, we propose a distributed routing algorithm to reduce power consumption of battery-powered devices by routing the communication through mains-powered devices whenever possible and consequently increasing the overall network lifetime. The proposed algorithm works on tree topologies supported by ZigBee and requires only minor modifications to the current specification. Our ns-2 simulation results showed that the algorithm is able to reduce the power consumption of battery-powered devices significantly with minimal communication overhead.

Keywords Power-source-aware · Routing · ZigBee · Tree-topology

1 Introduction

The ZigBee standard [38] defines a low-data rate wireless networking solution for interconnection of devices in a wireless personal area network (WPAN). The low-data rate requirement enables reduced complexity and very low power consumption, which are also the primary goals of ZigBee. The ZigBee standard is built on the IEEE 802.15.4 standard [20], which shares similar goals. ZigBee defines the application layer (APL) and the network layer (NWK), whereas IEEE 802.15.4 defines the medium access control layer (MAC) and the physical layer (PHY), as depicted in the protocol stack of Fig. 1. For the rest of the paper, ZigBee refers to the ZigBee and IEEE 802.15.4 standards as a whole, unless otherwise specified.

The PHY layer defines 16 channels in the 2,450 MHz band, 30 channels in the 915 MHz band, and three channels in the 868 MHz band [20]. Depending on the band, the devices can communicate with data rates of 250, 100, 40, and 20 kbps. The MAC layer controls access to the radio channel using the carrier sense multiple access with collision avoidance (CSMA/CA) mechanism. An optional superframe structure can be used to coordinate the channel access. A superframe, which is bounded by network beacons, can possibly include contention and contention-free access periods (CAP and CFA) as well as an inactive period. CFA periods can be assigned to time- or bandwidth-critical applications. On the other hand, inactive periods can be exploited to reduce power consumption by switching off the radio transmitters.

The NWK layer enables data transfer between devices that are not in the communication range of each other through the use of intermediate devices, hence making multi-hop communication possible. Responsibilities of the NWK layer include starting a network, coordinating joining and leaving a network, routing, discovering one-hop neighbors, and storing neighbor information. Three types of devices are possible in a ZigBee network: Coordinator, router and end devices. Routers are capable of forwarding

M. Tekkalmaz (✉) · I. Korpeoglu
Department of Computer Engineering, Bilkent University, Ankara, Turkey
e-mail: metint@cs.bilkent.edu.tr

I. Korpeoglu
e-mail: korpe@cs.bilkent.edu.tr

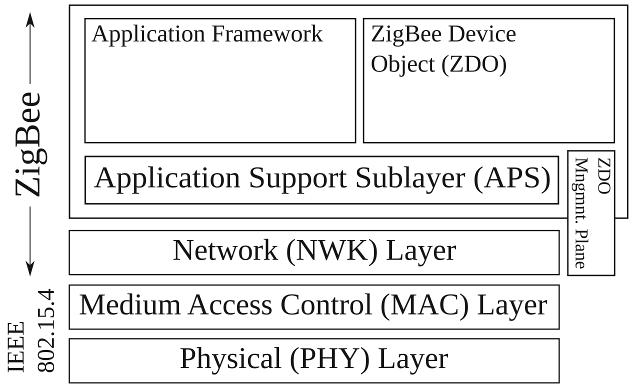

Fig. 1 ZigBee protocol stack

data on behalf of others and a coordinator is a router that starts the network and chooses key network parameters. Any device can connect to a router in the network, whereas, devices cannot connect to an end device, as the name implies. A ZigBee device is called a full functional device (FFD) if it can have a router role in the network and be a reduced functional device (RFD) otherwise. An RFD is usually limited in terms of its energy source (e.g. battery-powered), processing power, and memory capacity. Each ZigBee device has a universal 64-bit address and a 16-bit short address assigned when it connects to a network. Both tree and mesh topologies are possible in a ZigBee network. The ZigBee standard defines different address assignments and routing mechanisms for these topologies.

The APL layer of ZigBee consists of the application support sub-layer (APS) and the application framework. Responsibilities of the APS include maintaining tables used to bind devices according to the services provided and needed, forwarding between bound devices, fragmentation, reassembly, and reliable data transport. The application framework contains the ZigBee device object (ZDO) and manufacturer-defined application objects. ZDO defines the role of the device in the network, such as coordinator or end device, discovers application services, and manages service bindings.

As mentioned earlier, ZigBee targets low-data rate as well as low-duty cycle applications. Such applications include, but are not limited to, a wide range of control and monitoring applications such as building automation, industrial control, and sensor networks. A typical deployment site is likely to have battery-powered and mains-powered (i.e. AC-powered) devices coexisting: mains-powered devices would be preferred wherever possible in order to reduce maintenance costs, and battery-powered devices would be preferred where installing power lines is costly or practically impossible.

Two different routing schemes are specified in the ZigBee standard. One is hierarchical tree routing and the other is a modified version of ad hoc on-demand distance vector routing. In hierarchical tree routing, packets are routed according to the parent-child relationships established during ZigBee topology formation and distributed address assignment.

In this paper, we propose a power-source-aware routing algorithm, PSAR, for tree topology ZigBee networks. PSAR is based on hierarchical tree routing and simply aims at reducing the power consumption of battery-powered devices and consequently increasing the lifetime of the network. The basic approach to achieving this is to route the network traffic through mains-powered devices instead of battery-powered devices as much as possible. When routing in tree topology networks, because there is a single path between any two devices, the only way to reduce the burden on the battery-powered devices is to modify the existing network topology by disconnecting and reconnecting some devices to reduce traffic flow through the battery-powered devices.

PSAR requires only minor modifications to the current ZigBee protocol specification and minimal additional messaging, which keeps the overhead of the algorithms at a minimum. The simulation results showed that the average traffic on battery-powered devices was reduced by up to 50%, while there was no significant increase in the average path length between devices (hence neither in the total traffic load of the network) due to the topology changes.

The remainder of this paper is organized as follows: In the next section, related previous studies are summarized. The distributed address assignment scheme of the ZigBee standard, which is important for hierarchical routing, is given in Sect. 3, which is followed in Sect. 4 by a detailed description of our proposed routing scheme, PSAR. In Sect. 5, simulation results are presented. Finally, in Sect. 6, conclusions are given.

2 Related work

In the literature, co-existence of mains- and battery-powered devices in ZigBee wireless sensor networks is overlooked in general. As briefly mentioned in Sect. 1, there are various application scenarios in which deployment of devices with different types of power-sources is possible. For example, in home and building automation and industrial control applications [13–16, 37], continuous supply of power from the power grid is probably available within the facility that the ZigBee devices are deployed. In such a deployment site, some of the devices can benefit from the mains power. Home security and surveillance, being another application area of ZigBee technology [7, 18], can provide an environment in which mains-powered ZigBee devices can be deployed. Additionally, as presented in [6], ZigBee is an emerging communication technology in body area networks (BANs), especially for inter-BAN communication. Considering various uses of BANs, ZigBee access points, which are used to form an infrastructure, can be backed-up with continuous power-source from the powerline. ZigBee already provides a Smart Energy profile [38] and it can take an important role in efficient use of energy as in power management applications [31] by monitoring and reporting the energy usage of appliances. In such a setting, it would be possible to benefit from the power infrastructure that the ZigBee devices are connected to, for monitoring purposes.

There are several studies on increasing energy efficiency, hence the lifetime, of ZigBee networks. Baronti et al. [2] provides a survey of ZigBee networks as sensor networks and includes a section on energy efficiency. As presented in [2], energy-efficiency related approaches are realized in different layers of the protocol stack.

Suarez et al. [33] replace the MAC protocol of ZigBee with X-MAC. Cho et al. [10] adapt the beacon intervals dynamically based on the arrival rate of packets in order to increase the sleep time of the nodes. In a similar study, Kim et al. [21] presents the impact of adaptive superframe duration as well as of beacon interval. Li et al. [23] exploit multiple sleep/wake-up schedules as opposed to the single beacon interval of ZigBee.

Piccunelli et al. [29] present a strategy to build a routing tree based on a cross-layer cost function incorporating remaining energy, channel quality, and number of hops. Similarly, Boughanmi et al. [3] use a cost function in order to satisfy the energy and delay constraints of the paths to be used. At the path discovery phase of ZigBee, this modified function is used. Unlike studies in [29] and [3], Peng et al. [28] use the two ZigBee routing methods presented in the ZigBee standard as they are, but choose one of them according to the data service requiring routing functionality. In some studies, such as [19] and [32], multi-path routing is exploited in order to prolong the network lifetime.

In some studies, topology control is also applied in ZigBee networks to increase the network lifetime. Ma et al. [24], for example, proposes an algorithm to construct network topologies with a small number of coordinators while still maintaining network connectivity. The average duty cycle is reduced and the battery life is prolonged by reducing the number of coordinators.

As stated before, there are two different routing mechanisms (i.e. hierarchical tree routing and modified ad hoc on-demand distance vector routing—AODV) specified in the ZigBee standard and there are several studies analyzing and comparing these mechanisms [11, 25, 34]. Cuomo et al. [11] show that hierarchical tree routing performs better than AODV in terms of packet loss, energy consumption, and delay. Hierarchical tree routing exploits the information exchanged during the topology formation to achieve its superior performance. Although hierarchical tree routing is superior in certain scenarios, AODV provides a more generic routing solution, especially in relatively dynamic networks. Furthermore, hierarchical tree routing uses relatively longer paths compared to AODV. Studies in [17] and [22] make use of neighboring nodes, which are neither parents nor children of the current node, to enhance hierarchical tree routing by shortening the paths.

The main difference between the studies mentioned so far and the study presented in this paper is that PSAR distinguishes between mains- and battery-powered devices in order to modify the topology. Unlike residual energy, power-source information does not change over time. Hence, messaging required to share the energy levels is eliminated. Furthermore, studies that propose routing strategies are generally for mesh topology networks whereas our algorithm focuses on tree topologies. Hence, our algorithm makes use of advantages provided by hierarchical tree routing while eliminating the inefficient battery usage due to relatively longer paths.

In a recent study, Wang et al. [35] apply a pricing approach to form an energy efficient tree topology for ZigBee networks. In their study, both priority and energy of the devices are considered while forming parent-child relations. Although PSAR does not consider communication priorities of the devices, it can dynamically reconfigure the tree topology according to the changing traffic characteristics of the network. This is especially important if the communication characteristics of the network is unknown beforehand. In Radeke et al. [30], proposes a method that reconfigures the topology of the ZigBee network. The aim of the reconfiguration in [30] is to prevent exhaustion of address space, whereas the main purpose of PSAR is to increase network lifetime depending on the traffic demands. Different from the study in [30], PSAR also moves subtrees as a whole instead of a single node at a time. Furthermore, PSAR considers address changes after a reconfiguration occurs.

As far as energy efficiency in wireless sensor networks is considered, depending on the characteristics of the node and network properties and applications, a wide variety of approaches can be applied to consume less energy. For example, if data fusion and aggregation is applied in intermediate nodes, transmission times of aggregated data can be optimally scheduled [36]; if locations of the sensor nodes are known, energy efficient geographical routing strategies can be applied as in [8]; if use of a single sink causes bottlenecks, multiple sinks can be deployed whenever possible [26]; if there are mobile agents that can move through the region to collect data, efficient itineraries can be planned for the mobile agents [9]; and if devices support sleeping modes, they can be scheduled to go into sleep mode by using various energy-efficient sleep scheduling algorithms [5]. The difference of PSAR from all these various approaches for energy efficiency is that PSAR considers node heterogeneity in terms of energy sources while obtaining and updating a routing plan. Besides, it supports not only nodes-to-sink communication (convergecast), but also node-to-node (peer-to-peer) communication. Additionally, it is designed considering the constraints and requirements of a released network layer specification, i.e., the Zigbee Specification.

3 ZigBee address assignment

There are different mechanisms for address assignment depending on the topology (i.e. tree or mesh) of the ZigBee networks. In tree topology ZigBee networks, there are two alternatives for the network address assignment. In one of the alternatives, address assignment is left to the next higher layer. In the other alternative, the specification defines a distributed address assignment mechanism.

According to the distributed address assignment mechanism, every potential parent is assigned a finite block of network addresses. Each parent later assigns one (if the child is an end device) or more (if the child is router-capable, and therefore a potential parent) of these addresses to the devices connected to it. The coordinator of a network determines the maximum number of children that a parent can have, which is denoted by *Cm*. Of these children only *Rm* of them can be router-capable. Every device has a depth, *d*, which is the minimum number of hops to the ZigBee coordinator (i.e. the root of the tree). Maximum depth, *Lm*, of a tree network is also determined by the coordinator of that network. Given these values, the function $C_{skip}(d)$, which is actually the size of the address sub-block assigned to a router-capable device at depth $d + 1$, is computed as in (1) [38].

$$C_{skip}(d) = \begin{cases} 1 + Cm \cdot (Lm - d - 1), & \text{if } Rm = 1 \\ \frac{1 + Cm - Rm - Cm \cdot Rm^{Lm-d-1}}{1 - Rm}, & \text{otherwise} \end{cases} \quad (1)$$

A parent device assigns an address one greater than its own to its first router-capable child and the address of each such child is separated by $C_{skip}(d)$. The address of the *n*th end device, A_n, is computed as $A_n = A_{parent} + C_{skip}(d) \cdot Rm + n$, where $1 \le n \le (Cm - Rm)$ and A_{parent} is the address of the parent. Figure 2 depicts how the address space is used and redistributed at depth *d*.

Such a systematic address assignment mechanism enables a simple routing strategy. Any routing-capable device receiving a packet destined to an address *A* knows whether any of its children has address *A* or if *A* falls into the address sub-block of any of its children, in which case the packet is forwarded to the corresponding child. If no such child exists, then the packet is forwarded to the parent device. This routing strategy is called hierarchical routing and is applied in ZigBee tree topology. Although distributed address assignment eases the routing, one of its drawbacks is that whenever a device changes its parent, its and all of its descendants' network addresses need to change.

4 Power-source-aware routing

The basic strategy for our power-source-aware routing (PSAR) algorithm is to route the traffic through mains-powered ZigBee devices rather than battery-powered devices as much as possible. In a tree topology network there is a single simple path, hence, only one meaningful route between any two nodes. This means, once a topology is determined, no alternative route can be found in order to reduce the traffic routed by a battery-powered device. One possible solution is to modify the tree-based network topology dynamically depending on traffic demand so that the burden on the battery-powered devices is reduced.

Consider the ZigBee network given in Fig. 3, where *C* is the coordinator of the network, the nodes from R_1 to R_9 are the routers and *E*s are the end devices. In the figure, mains-powered routers are shown with solid lines, whereas battery-powered routers are shown with dashed lines. Assume that R_6 and its children have communication with R_3, R_4, and R_5. For the topology given in Fig. 3(a), R_6–R_3, R_6–R_4, and R_6–R_5 communications must follow paths R_2–C, R_2-C,

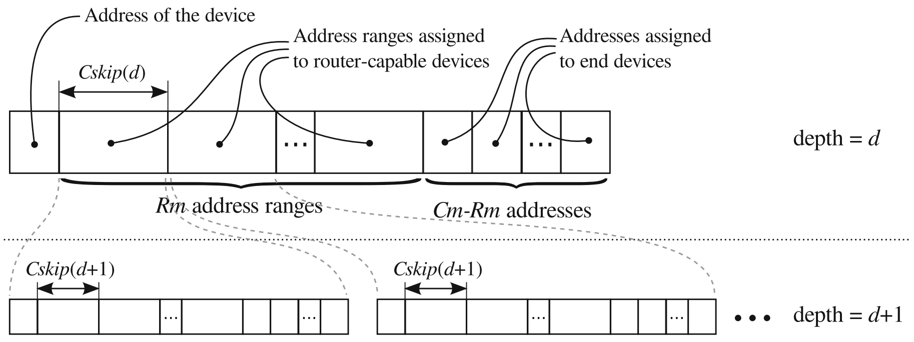

Fig. 2 ZigBee distributed address assignment

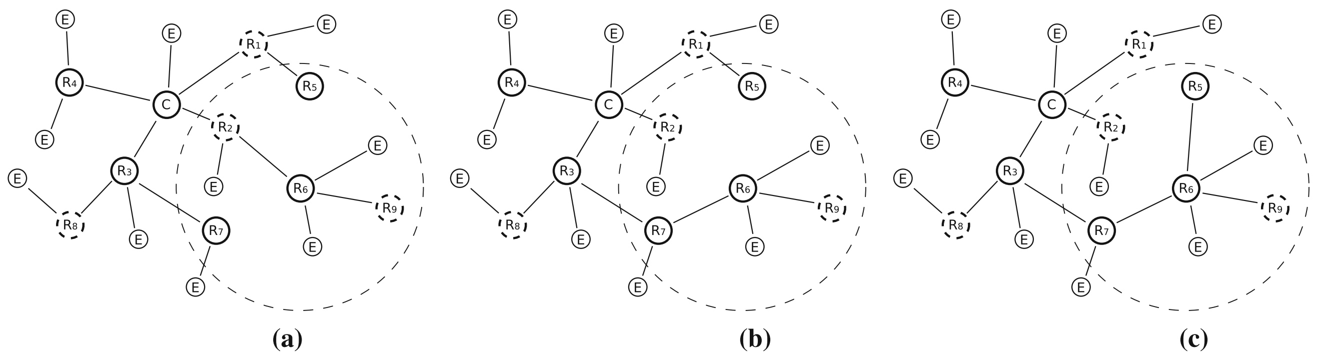

Fig. 3 Modifying the ZigBee tree topology in order to change communication paths

and R_2–C–R_1, respectively, meaning that battery-powered devices R_1 and R_2 are used as relay nodes. If R_6 is disconnected from R_2 and connected to R_7, as shown in Fig. 3(b), R_2 is eliminated from the communication paths, hence, the amount of traffic load on battery-powered devices is reduced. The next modification would probably be to disconnect R_5 from R_1 and connect it to R_6, as shown in Fig. 3(c), leaving all the communication paths free of battery-powered devices.

Unlike the example given above, it is not always possible to eliminate all the battery-powered devices on the communication paths due to constraints such as communication range, maximum number of children a router capable device can have and contradicting communication demands. Consider the simple topology given in Fig. 4 and assume that R_1 already has maximum number of children (i.e. C_m). If there is communication only between R_1 and R_4, disconnecting R_1 from R_2 and connecting it to R_3 reduces the total amount of data forwarded by battery-powered devices, since R_2 is eliminated from the communication path. But in the final topology R_3, which is a battery-powered device, still forwards data. If there is communication between R_1 and C, along with the R_1–R_4 communication, with a higher rate than it, the current topology is the optimum one as far as the traffic load on the battery-powered devices is considered. Therefore none of

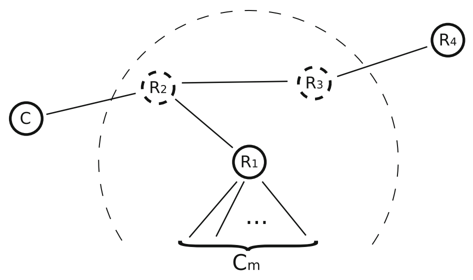

Fig. 4 A simple ZigBee tree topology with battery- and mains-powered devices

the battery-powered devices can be avoided from the communication paths.

Note that power-sources of the devices do not affect the general characteristics of the approach. For example, one or both of R_1 and R_4, given in Fig. 4, could be battery-powered, but disconnecting R_1 from R_2 and connecting it to R_3 would still reduce the traffic load on the the battery-powered devices. Additionally, intermediate devices, independent of their power-sources, can also react to reduce the amount of data forwarded by the battery-powered devices. For example, if R_1 had less than C_m children, R_3 could be connected to R_1 instead of R_2 and this reconfiguration would be beneficial especially for the second communication scenario, in which both R_1–R_4 and R_1–C pairs communicate.

Following a strategy like the one described so far reduces the amount of load on the battery-powered devices and it is possible to follow such a strategy provided that a device knows

- the amount of traffic it sends/receives/forwards,
- the paths for each such traffic for a given topology,
- the battery-powered devices on the paths, and
- the alternative devices that it can connect to.

Fortunately, a router-capable device in a ZigBee network can obtain all this information with minimal or no overhead in terms of network communication. The next section describes how to obtain and use this information to modify the topology.

4.1 The algorithm

As explained earlier, in ZigBee tree networks, a router-capable device relays all the traffic between its descendants and the rest of the network. Hence, for such a device it is possible to monitor the source, the destination, and the rate of each flow it forwards.

Thanks to the distributed address assignment mechanism of ZigBee, it is possible just by local computation to find the path between any two devices whose addresses are known, meaning that no communication overhead such as route discovery is required. Although not described in the ZigBee specification, we provide a way to compute the path from a device with address A_s to another device with address A_d in Algorithms 1 and 2. Algorithm 2 is used by Algorithm 1 to compute the path from the network coordinator to an arbitrary ZigBee node A in the network. Line 5 of Algorithm 2 computes the child of parent p, having node A as a descendant, where $C_{skip}(d)$ is calculated as in (1). Hence, starting from the root (lines 1 and 2), at each iteration of the while-loop, the ancestor of node A at depth $d + 1$ is found. It is easy to compute the path between any two nodes if the paths between those nodes and the root are known. As described in Algorithm 1, to compute the path between A_s and A_d, the common prefix, except for the closest common ancestor, is removed from the paths. Then one of the paths is reversed and concatenated to the other.

Unlike the path computation, determining the types of power sources of the devices requires additional communication, as this information does not generally exhibit a predictable pattern. One obvious way to collect this information is to let each battery-powered device register itself to the coordinator and let any device query this information whenever required.

Having this information, a router-capable device can now compute the total load on the battery-powered devices of the network due to communication between its descendants and rest of the network, as in Algorithm 3. Note that the battery-powered devices are implicitly obtained from the coordinator on line 4 of the algorithm. In the current form, the battery-powered devices are queried independently for each path. A more efficient approach would be to have a single query for the union of all paths, as they probably contain many common devices, but for the sake of simplicity, the algorithm is described in this way.

The method for finding the load on the battery-powered devices from the descendants of a router-capable device for the current topology, is described so far. In order to modify the topology, it is required that the node learns about alternative neighboring devices that can be connected to and computes the possible loads on the battery-powered devices in the alternative topologies. ZigBee provides neighbor discovery mechanisms, making it possible to determine whether other devices are in the communication range. Due to the distributed address assignment mechanism of ZigBee, it is also possible to compute the paths, obtain the battery-powered devices on the paths, and find out the load on them if a certain alternative router-capable device is chosen as the new parent, in the same way described previously.

Algorithm 1 Path between two ZigBee devices

```
function PathBetween($A_s$, $A_d$)
1 : $path_s \leftarrow$ PathFromRoot($A_s$)
2 : $path_d \leftarrow$ PathFromRoot($A_d$)
3 : $lcp \leftarrow$ longest common prefix of $path_s$ and $path_d$
4 : $path \leftarrow A_s$
    + reverse of ($path_s - lcp$)
    + last address in $lcp$
    + ($path_d - lcp$)
    + $A_d$
5: return $path$
```

As described in Algorithm 4, a router-capable device can decide the best parent in terms of load on the battery-powered devices and change its parent if the best case differs from the current one. As a restriction, the new parent should have equal or less depth than the current one (line 5). Otherwise some descendants of the device might not get a network address since the address range assigned to a device decreases as its depth increases. Please note that this is a conservative approach. Event if the depth of a parent candidate is greater, it is possible to obtain a network address for all the descendants of the device if the depth of the deepest device is still less than or equal to Lm (i.e. maximum allowed depth of the tree) after the reconfiguration. Since the depth information of some descendants might be unknown for a device without additional communication, this conservative approach is preferred. On the other hand, even if the parent candidate satisfies the restriction on the depth, it might already have Rm router-capable children, meaning that it cannot accept any other router-capable child. Hence a device requires explicit permission of the parent candidate before reconfiguration (line 15).

Note that the term connection mentioned in the 6th and the 16th lines of Algorithm 4 refers to the connection of the device with all of its descendants as a subtree. As mentioned in Sect. 3, whenever a device changes its parent, its address and the addresses of all its descendants have to be changed as well. The descendant devices are informed about the old and new addresses of the device starting the reconfiguration (i.e. root of the subtree) and each descendant utilizes this information to compute the updated addresses of its parent, children, and itself as described in Algorithm 5. Next, updated addresses are used to re-join to the network by orphaning mechanism described in the ZigBee specification.

Algorithm 5 takes old (R_o) and new (R_n) addresses of a device R, and old address (D_o) of another device D as input and returns either the new address of D, if it is a descendant of R or $\perp$, otherwise. In line 2, the algorithm checks whether D is a descendant of R. If so, in the while-loop between lines 5 and 11, the location of D in the subtree is

Algorithm 2 Path from the coordinator

```
function PathFromRoot($A$)
1: $d \leftarrow 0$
2: $p \leftarrow$ coordinator
3: while $A \neq p$ do
4:     $path \leftarrow path + p$
5:     $p \leftarrow p + 1 + \lfloor (A - (p+1)) \div C_{skip}(d) \rfloor \times C_{skip}(d)$
6:     $d \leftarrow d + 1$
7: end while
8: return $path$
```

Algorithm 3 Load on battery-powered devices

```
1: $load \leftarrow 0$
2: for all forwarded traffic $T$ do
3:     $path \leftarrow$ PathBetween ($T_{source}$, $T_{destination}$)
4:     $n \leftarrow$ number of battery-powered devices on $path$
5:     $load \leftarrow load + n \times T_{rate}$
6: end for
7: return $load$
```

Algorithm 4 Reconfiguration of the topology

```
1: $load \leftarrow$ load in the current configuration
2: $n \leftarrow 0$
3: for all router-capable neighbor $N$ do
4:     $c_n.neighbor \leftarrow N$
5:     if $depth_N < depth$ then
6:         $c_n.load \leftarrow$ load if the device connects to $N$
7:     else
8:         $c_n.load \leftarrow \infty$
9:     end if
10:    $n \leftarrow n + 1$
11: end for
12: sort $c$ in ascending order w.r.t. $load$ values
13: $i \leftarrow 0$
14: while $i < n$ and not connected to a new parent do
15:     if $c_i.load < load$ and router-capable child count of
        $c_i.neighbor < Rm$ then
16:         connect to $c_i.neighbor$
17:     end if
18:     $i \leftarrow i + 1$
19: end while
```

traced starting from the root address R_o and using its old address D_o and at each iteration of the loop, new address D_n of D is updated according to this location information given that the new root address is R_n.

Since the network addresses of the devices change, the rest of the network should be informed about these address changes in order to route the data packets to the correct devices. The new addresses are also required by the devices to update their power-source information caches in which power-source information is stored along with the network addresses in order to compute the load on the battery-powered devices whenever required. To disseminate this information, only the address change of the device starting the reconfiguration is broadcast in the network. Receiving the old and new addresses of the root of the subtree, which consists of devices whose addresses are updated, a device checks all the addresses it is interested in to see whether they belonged to the subtree and if so updates them accordingly using the function given in Algorithm 5. Since successful dissemination of address updates has vital importance for the ongoing communications and power-source information, which is required for later reconfigurations, a reliable method for broadcast should be chosen. In our current implementation, we transmit broadcast messages at most three times and try to limit the number of retransmissions applying passive acknowledgement mechanism as the ZigBee specification suggests. Other methods such as the one presented in [12] can also be utilized.

As long as a network has a stable traffic characteristic, the network is expected to converge to a topology in which battery-powered devices are avoided as much as possible. Because at each reconfiguration, topology is modified in a way that the total load on the battery-powered devices are reduced and reconfigurations occur as long as better topologies are found in terms of load on the battery-powered devices. But the algorithm can handle changes in the traffic characteristics (e.g. communicating pairs, bandwidth requirements, etc.) since the routers constantly monitor the packets they forward and react accordingly. Therefore, new reconfigurations may take place in order to adapt to new situations.

Although PSAR is especially designed for applications, in which members of the network and their locations are expected to be rather stable, such as building automation and industrial control, network structure can still change due to mobile devices (e.g. remote controllers, actuators), death of battery-powered devices or addition of new devices. In such situations, PSAR depends on the underlying mechanisms defined in the ZigBee standard [38], such as join via orphaning or rejoin procedures and device or service discovery, for the connectedness of the network and continuation of the existing communication. Hence, such topology changes are transparent to PSAR and it reacts to them similar to the changes in traffic characteristics as explained in the previous paragraph.

4.2 Implementation

We implemented the PSAR algorithms presented in Sect. 4.1 fully and efficiently in ns-2 (version 2.31) simulation environment [1], on top of the IEEE 802.15.4 and ZigBee protocol stacks. A module implementing 802.15.4 was already in ns-2, and we utilized that with slight modifications done wherever required (e.g. to support network addresses in addition to device addresses). However, at the time we did our simulations, there was no publicly available ZigBee module for ns-2, hence, we implemented the required parts of the ZigBee standard (that is, address assignment, routing, broadcast, rejoining, etc.) and integrated them with ns-2.

During a reconfiguration, a subtree of devices disconnects from a parent and connects to another as a whole, preserving the topology within the subtree. Therefore the load on the battery-powered devices of the subtree remain the same before and after the reconfiguration. Hence, in our PSAR implementation, as an efficiency measure the 4th line of Algorithm 3, and in turn, Algorithms 1 and 2, is implemented to avoid paths from the current node (i.e. the node executing the algorithm locally) to its descendants. Such an implementation choice reduces the bandwidth required to obtain the power sources of the nodes on those paths. On the other hand, in order to preserve the consistency of the network, simultaneous topology reconfigurations are not allowed. Otherwise, nodes may try to connect to nodes which are actually in the middle of an independent reconfiguration. In the current implementation, permission of the coordinator is obtained to begin a reconfiguration and the coordinator allows only one reconfiguration at a time.

The following properties of PSAR facilitates its implementation: (a) it mostly utilizes existing commands specified in the IEEE 802.15.4 and ZigBee standards, (b) it can be implemented as an application layer entity with minor modifications on the protocol stack. Table 1 lists the existing commands directly used in the implementation of PSAR. The scheme, which makes use of the listed commands, can be implemented as an extension to ZDO or as an application object, both of which are in the APL layer. Since the ZDO or an application object cannot monitor packets forwarded at the NWK layer, a new interface should be added to NWK, probably through NLME-SAP, to make necessary information available to the algorithm. On the other hand, there are cases in which the algorithm residing at the APL requires direct access to some of the services provided by the lower layers, skipping the NWK (for the ZDO and the application object alternatives) or APS of APL (for the application object alternative). As an example, `MLME-SCAN` is used by the NWK only if the device is currently not connected to a network, whereas the algorithm needs to discover nearby devices even if it is already part of a network. Rest of the control packets required by the algorithm, such as parent candidate permission requests/responses and dissemination of updated addresses, are implemented as part of the application layer protocol. Therefore, these messages are transferred by standard data exchange mechanism provided by NWK through `NLDE-DATA` command. Although standard commands are used during reconfigurations, at certain points additional implementation is required to satisfy preconditions of these commands. For example, in order to reconstruct the subtree using orphaning mechanism after parent-child relations are broken (using `NLME-LEAVE`), the neighbor tables are updated to reflect the parent-child relations and new network addresses beforehand, using the information obtained via previous control messaging.

As a summary, PSAR can be implemented as an application layer protocol, which mostly utilizes services provided by the lower layers of the ZigBee protocol stack. Most of these services are already accessible from the application layer, but some of them, which are implemented as part of the ZigBee specification, should be made available to the application layer. PSAR does not require modifications on the existing message structures, hence ZigBee devices with and without PSAR implementations are expected to be interoperable as far as the current ZigBee specification is considered. On the other hand devices without PSAR implementation cannot take part in topology reconfiguration defined by PSAR.

Algorithm 5 Updated address of a device

```
function ObtainUpdatedAddress ($R_o$, $R_n$, $D_o$)
1: $D_n \leftarrow \perp$
2: if $R_o < D_o$ and $D_o < (R_o + C_{skip}(depth_{R\_o} - 1))$ then
3:     $D_n \leftarrow R_n$
4:     $A_o \leftarrow R_o$
5:     while $A_o \neq D_o$ do
6:         $skip_o \leftarrow C_{skip}(depth_{A_o})$
7:         $skip_n \leftarrow C_{skip}(depth_{D_n})$
8:         $index \leftarrow \lfloor (D_o - (A_o + 1)) \div skip_o \rfloor$
9:         $A_o \leftarrow A_o + 1 + index \times skip_o$
10:        $D_n \leftarrow D_n + 1 + index \times skip_n$
11:    end while
12: end if
13: return $D_n$
```

4.3 Analysis

One of the aims of the ZigBee specification is to make the design and production of low-cost devices possible. Reducing the complexity of the hardware is an important part of this goal and software running on such reduced hardware is required to have low memory and processing power demands. First part of this section presents an asymptotic analysis of memory and processing power

Table 1 802.15.4 and ZigBee commands utilized in the implementation of the proposed algorithm

Command	Specified in	Purpose
`MLME-SCAN`	802.15.4	Used for discovering other devices in the communication range
`NLDE-DATA`	ZigBee	Used for transferring PSAR control packets
`NLME-LEAVE`	ZigBee	Used for disconnecting a subtree from the network
`NLME-JOIN`	ZigBee	Used for reconnecting all the nodes in the subtree. Since preserving the topology of the subtree is desired, rejoin through orphaning procedure is applied
`NLME-DIRECT-JOIN`	ZigBee	Used for preparing the candidate parent for the new child node
`Power_Desc_store_req` and `Power_Desc_store_rsp`	ZigBee	Used for storing the power source information of the devices in the coordinator
`Power_Desc_req` and `Power_Desc_rsp`	ZigBee	Used for retrieving the power source information of the devices from the coordinator

requirements of PSAR and its possible implementations. In the second part of the section, message complexity of PSAR is discussed.

First of all, the algorithm requires monitoring and keeping track of the forwarded traffic. For each communicating pair whose traffic is forwarded (in other words, for each traffic flow forwarded), the network addresses and the associated bandwidth requirements should be stored and updated. There are several options for storing this data, each having advantages and disadvantages in terms of memory (i.e. space complexity) and processing power (i.e. time complexity) requirements. The most straightforward implementation is to use a list, whose space complexity is $O(m)$ where m is the number of communicating pairs to be tracked. On the other hand, for each data packet forwarded, the list should be sequentially searched and the corresponding element should be updated, which has a time complexity of $O(m)$. An alternative, as preferred for the implementation of the simulation, is to use a binary search tree (BST). A BST has the same space complexity (i.e. $O(m)$) as the list implementation, and although it has a larger overhead per communicating pair to be tracked, the gain is in the $O(\log m)$ search time. Other options are also possible, all of which can be used for tracking the forwarded traffic, such as a sorted array with $O(m)$ space and $O(\log m)$ time complexity but costly maintenance as new communicating pairs arise, or a hash map with amortized $O(1)$ time complexity but possibly higher space complexity.

Having the communicating pairs and associated bandwidth requirements, nodes can decide whether a reconfiguration is necessary. As shown in Algorithms 1, 2, and 3, for each communicating pair the path between them should be computed. This leads to an average time complexity of $O(m\log n)$, assuming the depth of the tree-shaped network is bounded by $O(\log n)$ on average, where m is the number of communicating pairs whose traffic is forwarded by the current node and n is the total number of nodes in the network. In the worst case, there can be at most $O(n^2)$ flows (communicating pairs) going over a node of the network. This is, however, quite a loose upper bound. The paths can be computed each time they are required, as in the current implementation of the simulation, or cached, which increases the memory overhead and is probably not preferable. Once the paths are known, each node should be tested against its power source in order to compute the load on the battery-powered devices (see Algorithm 3, lines 4 and 5). Assuming the power source information is cached after it is obtained from the coordinator, in order to prevent unnecessary communication, the power source of the device on the computed paths can be searched from this cache. The cache can be implemented as a BST or hash map in order to favor time complexity or as a simple list in order to favor space complexity. Considering the possible number of unique nodes on the paths, the power source cache is implemented as a BST in the simulations, meaning that in the current implementation load computation has a time complexity of $O(m\log^2 n)$.

Until now possible space and time complexities of PSAR depending on the implementation alternatives have been presented. PSAR also requires extra control messaging as described in the previous sections and this part analyzes this messaging overhead asymptotically. Let n be the total number of nodes in the network and k be the number of nodes in the subtree to be connected to another node in the network. There are two groups of messages: one is exchanged once in a lifetime of a network and the other is once per reconfiguration. Messaging required to register power-source information of a device when it connects to a network and to query this information to compute the load on the battery powered devices belongs to the first group. $O(n\log n)$ messages are exchanged for both registration and querying, assuming a tree depth of $O(\log n)$. Since the ZigBee networks are expected to have long lifetimes, overhead of this group of messaging is considered to be negligible. On the other hand in each successful reconfiguration attempt $O(\log n)$ messages are exchanged to inform PAN coordinator about the start and end of a reconfiguration, $O(\log n)$ messages are exchanged to have permission of the parent candidate, $O(k)$ messages are exchanged for network leave and network join operations, and finally $O(n)$ messages are exchanged to inform the network about the address change of the device which initiates the reconfiguration. Hence neglecting the initial one-time overheads and considering $k < n$, the algorithm requires $O(n)$ messages per reconfiguration.

5 Simulation results

This section presents simulation results illustrating the performance of PSAR. In the simulations, several parameters are fixed. The communication band is set to 2,450 MHz and *Cm*, *Rm*, and *Lm* values are 6, 6, and 6, respectively. Furthermore, the superframe structure is not applied and all the devices in the network are chosen to be FFD. On the other hand, several parameters are changed in order to observe the impact of different conditions on the performance of the algorithm. These parameters, along with their chosen values, are network size (10, 40, 70 devices), density (one device per 24, 16, and 8 m^2), battery-powered device ratio (10, 20, …, 90%) and ratio of data flow count to total number of devices (10, 30, and 50%). Note that we use node count, not the number of all possible pairs, while limiting the traffic flows. Otherwise, the number of flows (pairs) would be excessive. For each combination of aforementioned parameters, the results are averaged across 100 simulations (disconnected topologies due to communication range and the orphan problem [27] are eliminated), in each of which node locations, battery-powered devices, and communicating nodes were determined pseudo-randomly, as described in [4].

Traffic flows are constant bit rate (CBR) flows with 2-s data generation interval and a random packet size between 2 and 50 bytes. In each simulation, 120 min of communication is simulated and each device is configured to check for a possible reconfiguration every 20 min with a randomization of ±20 s, in order to prevent simultaneous reconfiguration attempts. Another alternative for triggering the algorithm on a device is to wait until a significant change occurs in the traffic observed by that device. In the majority of the simulation results presented in this section, the communicating pairs are fixed (i.e. static traffic) for a simulation run. But the results in which communicating pairs change during a simulation run (i.e. dynamic traffic) are also given in order to present how PSAR copes with the dynamic traffic scenarios. Please note that, the static traffic case can be interpreted as a stable portion of a longer and dynamic (in terms of communication demands and device arrivals and departures) traffic case.

Currently, to the best of our knowledge, there is not a similar study with PSAR in the literature that is adapting the tree topology dynamically with respect to the traffic demand to reduce the load on battery-powered devices. Therefore, simulations are repeated in the presence and the absence of PSAR (i.e. using base ZigBee tree routing reported in the specification), keeping all the remaining parameters intact for both cases. Change (percent reduction) in the following metrics are measured in order to evaluate the performance of the algorithm: Total amount of data forwarded by all the battery-powered devices, standard deviation of the forwarded data by the battery-powered devices, and average path lengths between communicating devices. Percent reduction is defined as in (2). Apart from the above, packet drops and communication overhead due to PSAR are also measured.

$$\text{Reduction} = \frac{\text{Value without PSAR} - \text{Value with PSAR}}{\text{Value without PSAR}} \times 100 \tag{2}$$

Figures 5, 9, 10, 11, and 12 depict the reduction in total traffic load of the battery-powered devices, the reduction in the standard deviation of the traffic loads of the battery-powered devices, the reduction in average path lengths between communicating node pairs, the packet drop rates, and control packet ratio for the static traffic case, respectively. In these figures, the columns present simulation results for the network sizes of 10, 40, and 70 devices and the rows present simulation results for the networks with densities of one device per 24 m^2, 16 m^2, and 8 m^2. The network size increases from left to right and the device density increases from top to bottom. In each graph, values for three different communicating pair ratio (i.e. ratio of communicating pair, or traffic flow, count to the total node count) cases are given. Figure 8 presents the reduction in total traffic load of the battery-powered devices for the dynamic traffic case.

Before analyzing the effect of different parameters on the performance of PSAR, let us show how PSAR helps increasing the lifetime of a network. In a network that is composed of both mains- and battery-powered devices, lifetime of the network directly depends on the lifetime of the battery-powered devices. As shown in Fig. 5, if PSAR is applied, although additional control packets need to be forwarded, traffic forwarded by a battery-powered device is reduced on the average, which means each battery-powered device has a longer expected lifetime. This result does not necessarily mean that the lifetime of the network is increased, since some battery-powered devices might die much earlier than the case without PSAR (although on the average the battery-powered devices live longer), leaving some portions of the network unreachable. But Fig. 9 shows that if PSAR is applied, standard deviation of the

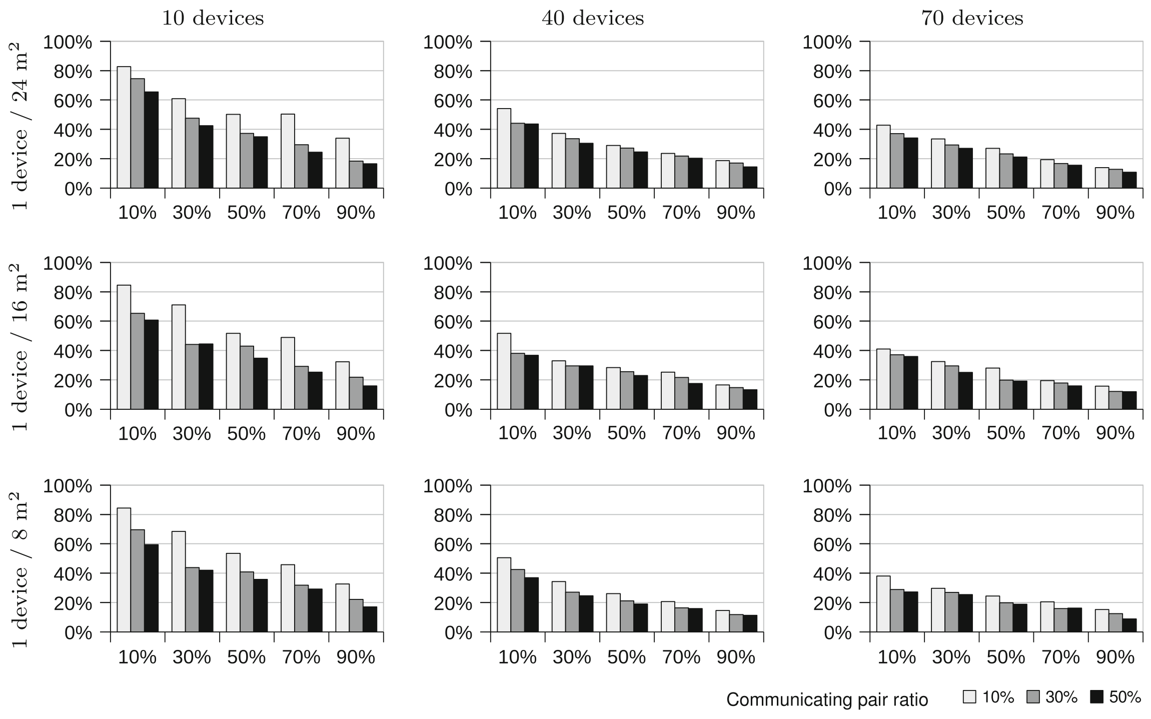

Fig. 5 Percent reduction in traffic load on the battery-powered devices (x-axis: battery-powered device ratio, y-axis: percent reduction)

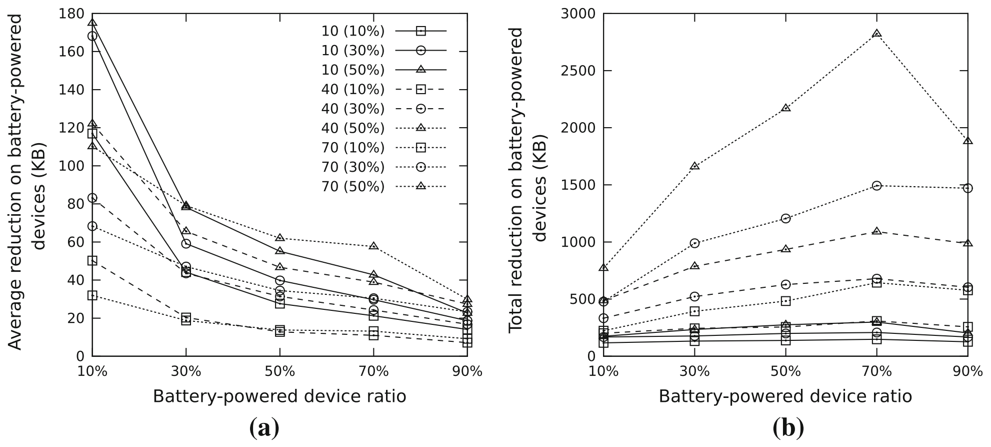

Fig. 6 **a** Average and **b** total reduction in the load on the battery-powered devices for the network sizes of 10, 40, and 70 devices and communicating pair ratios of 10, 30, and 50%

traffic forwarded by the battery-powered devices is also reduced, meaning that the lifetime of the battery-powered devices are distributed more evenly. Hence we claim that lifetime of the network increases, since the lifetime of each battery-powered device increases.

As far as the percent reduction in total traffic load on the battery-powered devices are concerned, values as high as 80, 50 and 40% are observed for the network sizes of 10, 40 and 70, respectively (see Fig. 5). But as the battery-powered device ratio increases, the percent reduction in total traffic load decreases to values as low as around 10%. There are two obvious reasons for this decrease in the traffic load percent reduction. First, an increase in the number of battery-powered devices does not correspond to an increase at the same rate in the number of battery-powered devices avoided from the paths between communicating pairs, because it gets harder to find paths without battery-powered devices. Second, even if some of the battery-powered devices are avoided in networks with a higher battery-powered device ratio, the total load on all battery-powered devices is so high that the reduced amount of load does not have a comparatively significant value. Another observation is that as the network size increases, the percent reduction in traffic load on the battery-powered devices decreases. The reason is similar to the previous argument, that is, although the amount of traffic load avoided from battery-powered

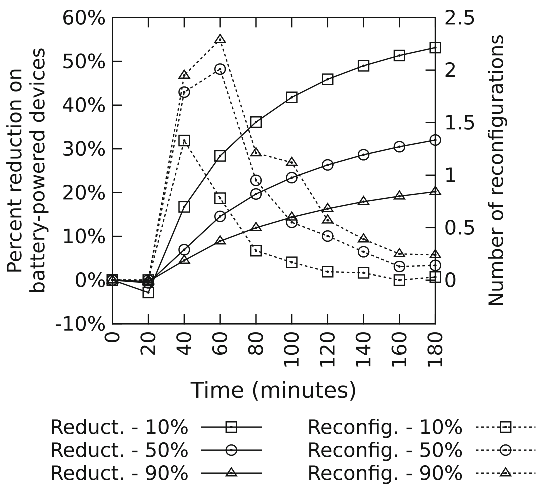

Fig. 7 Percent reduction in traffic load on the battery-powered devices and number of configurations over time for the network size of 40 devices and communicating pair ratio of 30%

devices does not change significantly, since the number of battery-powered devices (and therefore, the total load on them) increases, the significance of the avoided traffic load decreases.

Figure 6(a), (b) support these arguments: As the battery-powered device ratio increases, average reduction per battery-powered device decreases while the total reduction on all the battery-powered devices increases. Furthermore, larger networks have better average and total reduction values in bytes although they have worse percent reduction values since the avoided traffic does not keep up with the increase in the number of battery-powered devices. In Table 2, total amount of traffic (i.e. all traffic sent from the source nodes and forwarded by the intermediate nodes) for different network sizes and communicating pair ratios is given in order to compare with the values presented in Fig. 6.

In Figs. 5 and 6, reduction for the first 2 h of different networks are given. Differently in Fig. 7, percent reduction values until different time points from the beginning of the network is presented (e.g. y value which corresponds to the 60 in the x-axis is the percent reduction for 0–60 period). Figure 7 also depicts the number of successful reconfigurations for the last time period (e.g. y value which corresponds to the 60 in the x-axis is the number of reconfigurations for 40–60 period). As it can be seen from the figure, reduction increases over time although its rate decreases and tends to converge to a certain value. Additionally, for the first 20 min of the network lifetime, negative reduction values are obtained, since there have been messaging overhead on the battery-powered devices due to PSAR although there has not been any reconfigurations to reduce the amount of traffic forwarded by the battery-powered devices. Note that the number of reconfigurations peaks early in the network lifetime and decreases rapidly, meaning that the network topology converges rather quick considering a node can attempt for a reconfiguration once in every 20 min and simultaneous reconfigurations are not allowed. Also note that in Fig. 5 reduction values are given for the first 120 min of the network and Fig. 7 shows that the percent reduction continues to increase after this period, as long as the network traffic stays the same.

Figure 8 gives the similar set of results with the ones in Fig. 5 for the dynamic traffic case. In the simulations with the dynamic traffic scenario, the communicating devices are changed in the middle of the simulation period (i.e. around 60th min). As it can be observed from the figures, in the dynamic traffic case, the percent reductions are slightly less compared to the static traffic case. This is expected since the benefit obtained due to reconfigurations has effect for less amount of time in the dynamic traffic case. As the traffic characteristics change, current topology, which is the result of previous reconfigurations, would probably not be the optimal one, as far as the load on the battery-powered devices is considered. Since, recognizing the current traffic characteristics and adapting the topology accordingly take time, dynamic traffic patterns have negative impact on the performance of PSAR.

As shown in Fig. 9, PSAR is also able to decrease the standard deviation of the traffic load on the battery-powered devices. This result means that the load on the

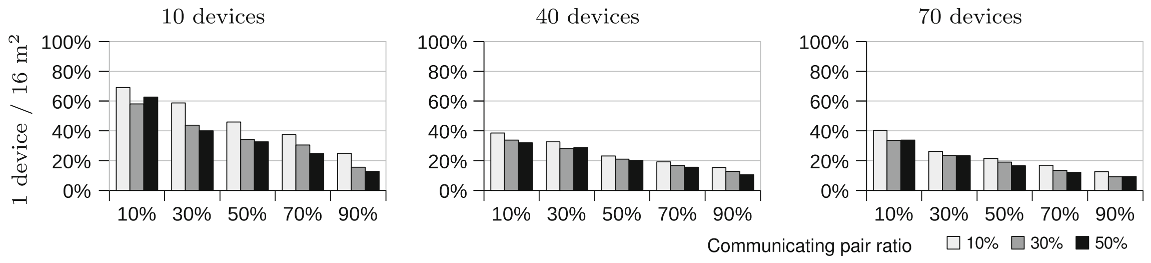

Fig. 8 Percent reduction in traffic load on the battery-powered devices for the dynamic traffic case (x-axis: battery-powered device ratio, y-axis: percent reduction)

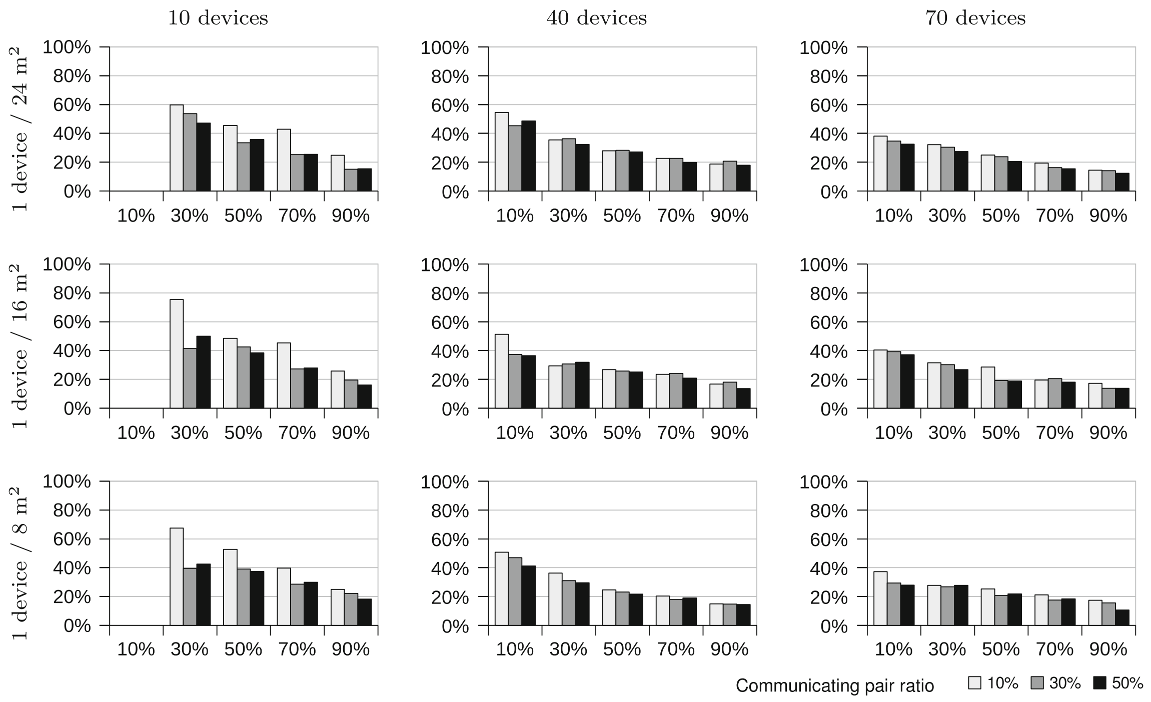

Fig. 9 Percent reduction in standard deviation of traffic load on the battery-powered devices (x-axis: battery-powered device ratio, y-axis: percent reduction)

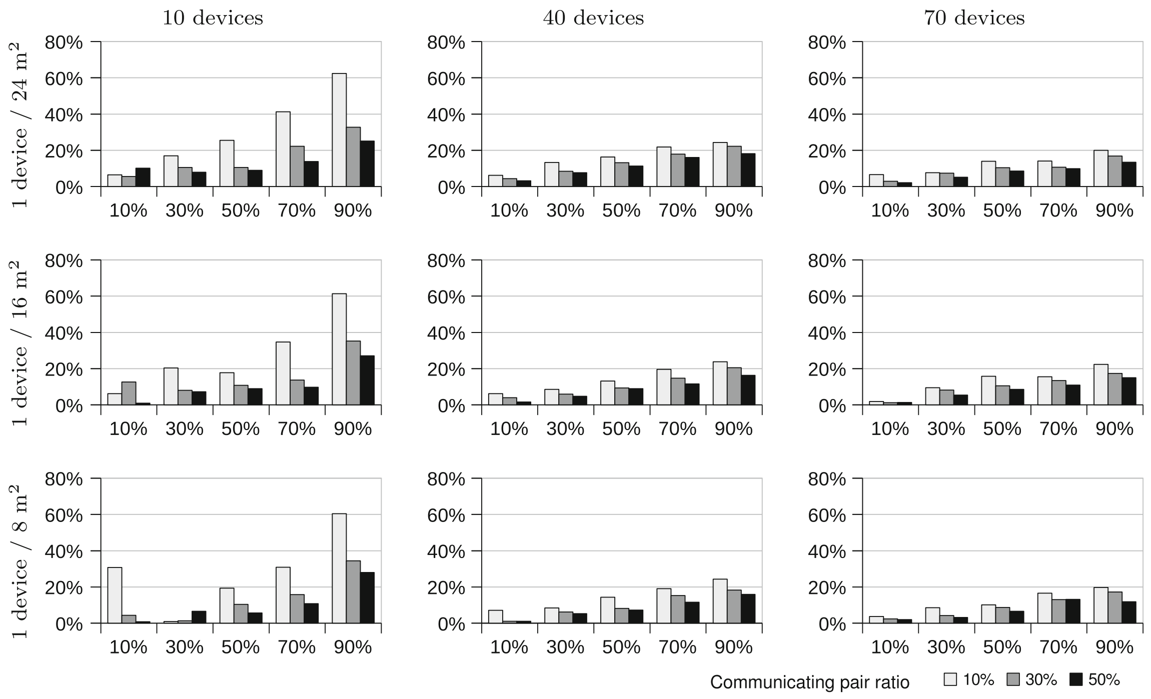

Fig. 10 Percent reduction in average path lengths between communicating devices (x-axis: battery-powered device ratio, y-axis: percent reduction)

battery-powered devices is not only reduced but also distributed more evenly, as stated earlier. The primary reason for the reduction in the standard deviation is that since the load on the most of the battery-powered devices are reduced or completely eliminated, the quantity of the differences is also reduced. The reduction in the standard deviation exhibits a similar characteristic with the reduction in the traffic itself, that is, the reduction in the standard deviation

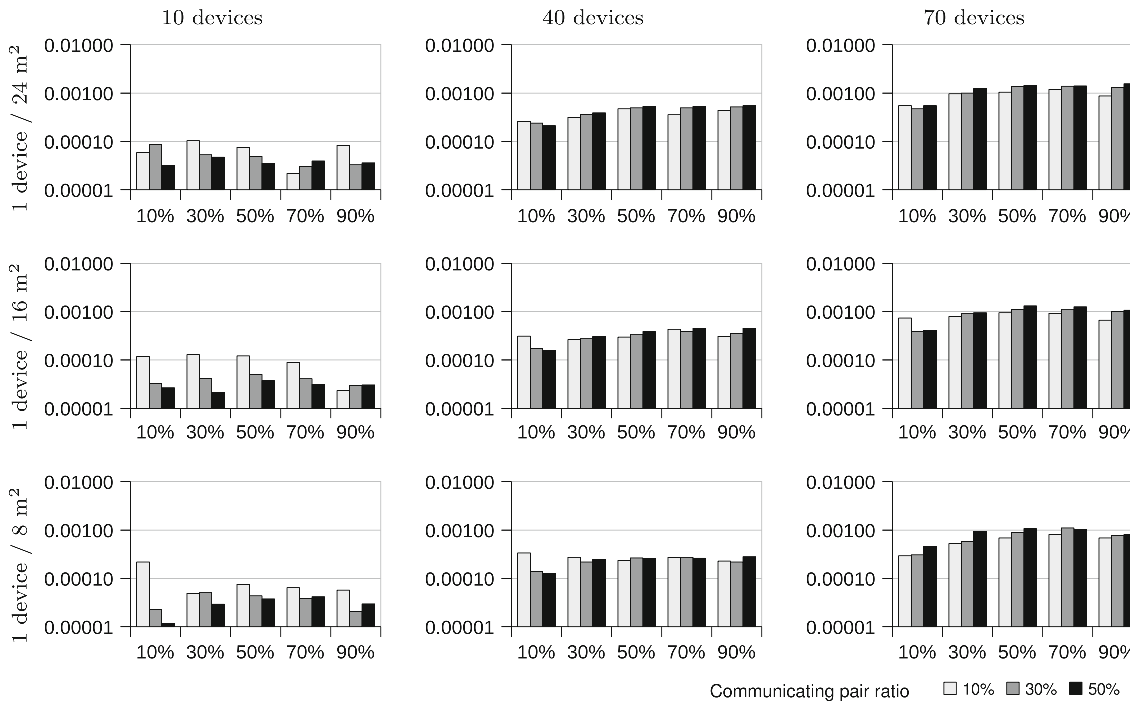

Fig. 11 Packet drop rate (x-axis: battery-powered device ratio, y-axis: drop rate in packets per second)

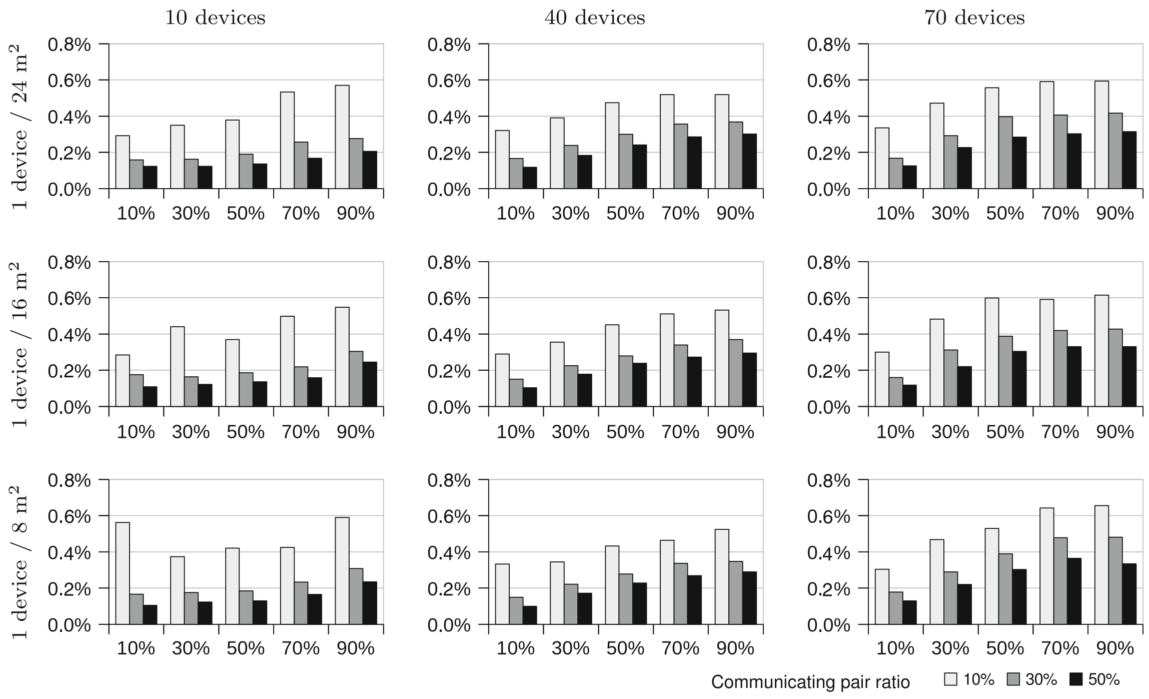

Fig. 12 Ratio of control packet traffic to data traffic (x-axis: battery-powered device ratio, y-axis: control packet ratio)

decreases from around 60% to below 20%. The reason for the decrease in the traffic load reduction described previously, largely applies to this case as well. The number of battery-powered devices avoided from the communication paths shows little change as the number of battery-powered devices increases, hence, the effect of the algorithm remains limited in the variation of the traffic load on them. As the network size increases, the reduction in standard deviation of traffic load on the battery-powered devices decreases, due to similar reasons given for the traffic load case.

Table 2 Total traffic in KB for different network sizes and communicating pair ratios

%	10 devices	40 devices	70 devices
10	523	2645	5,592
30	1,210	7,836	17,448
50	1,959	13,373	30,741

Experiments were designed to run on different device densities to observe the effect of device density on the effectiveness of PSAR. But as can be seen from the figures, neither the reduction in traffic load on the battery-powered devices nor the reduction in the standard deviation of the loads on the battery-powered devices are affected by device density significantly.

Although the primary concern of reconfigurations in PSAR is to reduce the traffic load on the battery-powered devices, it also helps to reduce the average path lengths between communicating devices, as shown in Fig. 10. There are two reasons for this side benefit. First, if there are more than one reconfiguration alternatives with equal reduction amounts, which is a rare case, then the one with a shorter path is preferred. Second, a node marks another node as a new parent candidate only if that node has equal or less depth value, reducing the average depth of the tree, hence its diameter.

As described in Sect. 4.1 once the network address of devices change due to reconfigurations, the address changes are advertised using a broadcast message. As a negative effect, between the time a destination changes its address and the corresponding source recognizes the change, the data packets are sent to the old address of the destination, which leads to packet drops. Hence, one of the aims of the experiments was to see the effect of the algorithm on the packet drops due to disconnections during the reconfigurations. As presented in Fig. 11 as the network size increases from 10 to 70 devices the packet drop rate increases from below 0.01% to around 0.1%. These results correspond to less than 1, 5 and 20 packet drops on the average for network sizes of 10, 40 and 70 nodes, respectively, given that 2 h of communication, CBR with 2-s intervals and 50% communicating pair ratio (i.e. 5, 20 or 35 data flows).

The communication overhead of PSAR was also observed in the experiments and the results are given in Figs. 12 and 13. The communication overhead of the algorithm is mainly due to registering the power source of the devices, query the power source of the devices, request connection from a parent candidate, inform the subtree about an upcoming reconfiguration, and inform the rest of the network about new network addresses after the reconfiguration. Not all the communication ends up with a successful reconfiguration, due to reasons such as not being a better alternative found after the power sources are queried

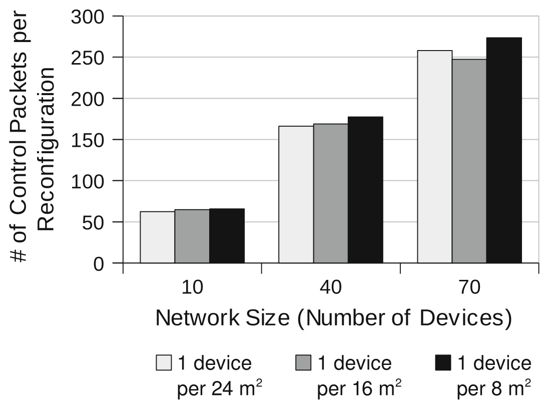

Fig. 13 Change in control packet count per reconfiguration with respect to network size

or because a parent candidate does not accept the new connection. Hence, two approaches are applied to measure the control packet overhead traffic due to PSAR. The first approach is to find out the ratio of control packet traffic to the actual data traffic (i.e. amount of control packet traffic divided by the amount of data traffic) and the second approach is to measure number of control packets per reconfiguration (i.e. total number of control packets divided by the total number of reconfigurations). As shown in Fig. 12, the control packet ratio is always below 0.7%. Note that the network size does not have an observable effect on the ratio, because the control packet traffic and the data traffic increase at the same rate as the network size increases. The communication overhead is approximately 60, 170, and 260 packets per reconfiguration for the network sizes of 10, 40, and 70 devices respectively, as depicted in Fig. 13. Hence the number of control packets per device per reconfiguration is around 6 for the network size of 10, while it is below 5 for the network sizes of 40 and 70. This is due to the less number of reconfigurations for the network size of 10 devices in which the effect of initial communication overhead per reconfiguration is higher.

6 Conclusions

In this paper we propose a distributed algorithm, PSAR, to reduce the traffic load on the battery-powered devices in tree topology ZigBee networks. The basic approach is to route the network traffic through mains-powered devices instead of battery-powered devices as much as possible. In order to achieve this, the topology must be modified as there is only a single path between any two nodes in a tree. New topology is decided by local computations with minimal communication to gather the required information. Simulation results showed that the reduction in traffic routed via battery-powered devices is as high as 80% in some cases. Simulation results also showed that PSAR reduces the standard deviation of the traffic load on the battery-powered devices so that the energy consumption is distributed more evenly among those devices. These benefits are obtained with insignificant communication overhead (due to control packets) and packet drops (due to disconnections during reconfigurations).

Acknowledgments This work is supported in part by the EU FP7-Program-ENV FIRESENSE Project, Contract Number 244088.

References

1. The network simulator. http://www.isi.edu/nsnam/ns/.
2. Baronti, P., Pillai, P., Chook, V. W. C., Chessa, S., Gotta, A., & Hu, Y. F. (2007). Wireless sensor networks: A survey on the state of the art and the 802.15.4 and ZigBee standards. *Computer Communications, 30*(7), 1655–1695.
3. Boughanmi, N., & Song, Y. (2008) A new routing metric for satisfying both energy and delay constraints in wireless sensor networks. *Journal of Signal Processing Systems, 51*(2), 137–143.
4. Camilo, T., Silva, J. S., Rodrigues, A., & Boavida, F. (2007). Gensen: A topology generator for real wireless sensor networks deployment. In *5th IFIP workshop on software technologies for future embedded and ubiquitous systems*.
5. Cao, Q., Abdelzaher, T., He, T., & Stankovic, J. (2005). Towards optimal sleep scheduling in sensor networks for rare-event detection. In *Information processing in sensor networks, 2005. IPSN 2005. Fourth international symposium on* (pp. 20–27).
6. Chen, M., Gonzalez, S., Vasilakos, A., Cao, H., & Leung V. C. (2011). Body area networks: A survey. *Mobile Networks and Applications 16*(2), 171–193.
7. Chen, M., González, S., Cao, H., Zhang, Y., & Vuong, S. (2010). Enabling low bit-rate and reliable video surveillance over practical wireless sensor network. *The Journal of Supercomputing* (pp. 1–14).
8. Chen, M., Leung, V. C. M., Mao, S., Xiao, Y., & Chlamtac, I. (2009). Hybrid geographic routing for flexible energy—delay tradeoff. *Vehicular Technology, IEEE Transactions on, 58*(9), 4976–4988.
9. Chen, M., Yang, L. T., Kwon, T., Zhou, L., & Jo, M. (2011). Itinerary planning for energy-efficient agent communications in wireless sensor networks. *Vehicular Technology, IEEE Transactions on, 60*(7), 3290–3299.
10. Cho, D. H., Song, J. H., & Han, K. J. (2006). An adaptive energy saving mechanism for the IEEE 802.15.4 lr-wpan. *Lecture Notes in Computer Science, 4138*, 38–46
11. Cuomo, F., Luna, S. D., Monaco, U., & Melodia, T. (2007). Routing in ZigBee benefits from exploiting the IEEE 802.15.4 association tree. In *IEEE international conference on communications, 2007. ICC '07.* (pp. 3271–3276).
12. Ding, G., Sahinoglu, Z., Bhargava, B., Orlik, P., & Zhang, J. (2005). Reliable broadcast in ZigBee networks. In *2005 2nd annual IEEE communications society conference on sensor and AdHoc communications and networks* (pp. 510–520).
13. Egan, D. (2005). The emergence of ZigBee in building automation and industrial control. *Computing and Control Engineering Journal, 16*(2), 14–19.
14. Gill, K., Yang S. H., Yao F., & Lu, X. (2009). A ZigBee-based home automation system. *Consumer Electronics, IEEE Transactions on, 55*(2), 422–430.
15. Gomez, C., & Paradells, J. (2010). Wireless home automation networks: A survey of architectures and technologies. *Communications Magazine, IEEE, 48*(6), 92–101.
16. Gutierrez, J. A. (2004). On the use of ieee 802.15.4 to enable wireless sensor networks in building automation. In *Personal, indoor and mobile radio communications, 2004. 15th IEEE international symposium on* (Vol. 3, pp. 1865–1869).
17. Ha, J. Y., Park, H. S., Choi, S., & Kwon, W. H. (2007). Ehrp: Enhanced hierarchical routing protocol for ZigBee mesh networks. *IEEE Communications Letters, 11*(12), 1028–1030.
18. Hou, J., Wu, C., Yuan, Z., Tan, J., Wang, Q., & Zhou, Y. (2008). Research of intelligent home security surveillance system based on ZigBee. In *Intelligent information technology application workshops, 2008. IITAW '08. International Symposium on* (pp. 554–557).
19. Huang, H. C., Huang, Y. M., & Ding, J. W. (2006). An implementation of battery-aware wireless sensor network using ZigBee for multimedia service. In *International conference on consumer electronics* (pp. 369–370).
20. IEEE Computer Society: Part 15.4. (2006). Wireless medium access control (MAC) and physical layer (PHY) specifications for low-rate wireless personal area networks (WPANs).
21. Kim, J. A., Jean, Y. H., & Park, H. S. (2006). Period assignment and optimal scheduling for ZigBee-based sensor networks. In *International joint conference SICE-ICASE, 2006* (pp. 1030–1035).
22. Kim, T., Kim, D., eun Yoo, N. P. S., & Lopez, T. S. (2007). Shortcut tree routing in ZigBee networks. In *2nd international symposium on wireless pervasive computing, 2007 (ISWPC '07)*.
23. Li, Y., Ye, W., & Heidemann, J. (2005). Energy and latency control in low duty cycle mac protocols. In *IEEE wireless communications and networking conference, 2005* (Vol. 2, pp. 676–682).
24. Ma, J., Gao, M., Zhang, Q., & Ni, L. M. (2007). Energy-efficient localized topology control algorithms in ieee 802.15.4-based sensor networks. *IEEE Transactions on Parallel and Distributed Systems, 18*(5), 711–720
25. Nefzi, B., & Song, Y. Q. (2007). Performance analysis and improvement of ZigBee routing protocol. In *7th IFAC, international conference on fieldbuses and networks in industrial and embedded systems, 2007* (Vol. 7).
26. Oyman, E. I., & Ersoy, C. (2004). Multiple sink network design problem in large scale wireless sensor networks. In *Communications, 2004 IEEE international conference on* (Vol. 6, pp. 3663–3667).
27. Pan, M. S., & Tseng, Y. C. (2007). The orphan problem in ZigBee-based wireless sensor networks. In *Proceedings of the 10th ACM symposium on modeling, analysis, and simulation of wireless and mobile systems (MSWiM '07)* (pp. 95–98).
28. Peng, R., Mao-heng, S., & You-min, Z. (2006). ZigBee routing selection strategy based on data services and energy-balanced ZigBee routing. In *Proceedings of the 2006 IEEE Asia-Pacific conference on services computing (APSCC '06)* (pp. 400–404).
29. Puccinelli, D., Sifakis, E., & Haenggi, M. (2006). A cross-layer approach to energy balancing in wireless sensor networks. *Lecture Notes in Control and Information Sciences, 331*, 309–324
30. Radeke, R., Marandin, D., Claudios, F. J., Todorova, P., & Tomic, S. (2008). On reconfiguration in case of node mobility in clustered wireless sensor networks. *Wireless Communications, IEEE, 15*(6), 47–53
31. Shah, P., Shaikh, T., Ghan, K., & Shilaskar, S. (2008). Power management using ZigBee wireless sensor network. In *Emerging trends in engineering and technology, 2008. ICETET '08. First International Conference on* (pp. 242–245).
32. Shillingford, N., Salyers, D. C., Poellabauer, C., & Striegel, A. (2007). Detour: Delay- and energy-aware multi-path routing in wireless ad hoc networks. In *International conference on mobile*

and ubiquitous systems: Networking and services, 2007. MobiQuitous 2007 (pp. 1–8).
33. Suarez, P., Renmarker, C. G., Dunkels, A., & Voigt, T. (2008). Increasing ZigBee network lifetime with x-mac. In *Proceedings of the REALWSN 2008 workshop on real-world wireless sensor networks* (p. 5).
34. Sun, J., Wang, Z., Wang, H., & Zhang, X. (2007). Research on routing protocols based on ZigBee network. In: *Proceedings of the 3rd International Conference on International Information Hiding and Multimedia Signal Processing. IIH-MSP 2007* (pp. 639–642).
35. Wang, J., Chen, M., & Leung, V. (2011). *Forming priority based and energy balanced ZigBee networks—a pricing approach*. Telecommunication Systems.
36. Yao, Y., & Giannakis, G. B. (2005). Energy-efficient scheduling for wireless sensor networks. *Communications, IEEE Transactions on, 53*(8):1333–1342
37. Zheng, L. (2006). ZigBee wireless sensor network in industrial applications. In *SICE-ICASE, 2006. International joint conference* (pp. 1067–1070).
38. ZigBee Standards Organization. (2006). *ZigBee specification*.

Author Biographies

Metin Tekkalmaz received his B.S. and M.S. degrees in computer engineering from the Bilkent University, Ankara, Turkey, in 2002 and 2004, respectively. Currently, he is a software engineer in ASELSAN Inc. and a Ph.D. student in the Bilkent University. His research interests include wireless ad hoc and sensor networks. He is a member of IEEE and ACM.

Ibrahim Korpeoglu received his Ph.D. and M.S. degrees in computer science from University of Maryland, College Park, USA, and B.S. degree in computer engineering from Bilkent University, Ankara, Turkey. He is currently an associate professor in the Department of Computer Engineering of Bilkent University. Prior to that, he has worked in several companies in USA including Ericsson, IBM T.J. Watson Research Center, Bell Laboratories, and Telcordia Technologies. His research interests include wireless ad hoc and sensor networks, mobile computing, peer-to-peer networks, and distributed systems. He is a member of ACM and a senior member of IEEE.