Flexible Address Configurations for Tree-Based ZigBee/IEEE 802.15.4 Wireless Networks

Li-Hsing Yen
Dept. Computer Science & Information Engineering
National University of Kaohsiung
Kaohsiung, Taiwan 811, R.O.C.
lhyen@nuk.edu.tw

Wei-Ting Tsai
Dept. CSIE
Chung Hua University
Hsinchu, Taiwan 300, R.O.C.
m09402042@cc.chu.edu.tw

Abstract

A number of IEEE 802.15.4 devices can be connected by a tree topology as proposed by ZigBee specification. Address configuration in tree-based ZigBee networks needs to assign every device a network address that uniquely identifies it from others, and such addressing should also assist routing. The addressing method recommended by the specification forces a static assignment that is coupled with node's location in the tree, resulting in an inflexibility in allocating addresses. This property may significantly decrease the ratio of addressable devices and cause routing detour. To alleviate the problem, this paper considers three alternatives that manage address space with flexibility but require additional storage in ZigBee routers. Performance evaluations indicate that proposed approaches provide different levels of tradeoff between the ratio of addressable devices and storage costs in ZigBee routers.

1. Introduction

IEEE 802.15.4 [1] is a standard for wireless Personal Area Networks (PANs), which comprise devices that are characterized by low data rate, short communication range, and low cost. Depending on their capabilities, these devices can be categorized into full function devices (FFDs) and reduced function devices (RFDs). FFDs are able to forward frames for other devices, while RFDs lack such capability. An FFD can initiate a PAN and act as the coordinator of the PAN. A coordinator can periodically broadcast beacon frames so nearby RFDs can discover it and thereby join the PAN, forming a star topology.

ZigBee specification [2] extends the basic star topology of an IEEE 802.15.4 PAN to a tree or mesh. In a tree topology, the root (called ZigBee Coordinator; ZC) and all internal nodes (called ZigBee Routers; ZRs) are FFDs, while RFDs can only be leaf nodes called ZigBee End Devices (ZEDs). When a ZR or ZED joins the network, it must be assigned a network address that is unique in the tree. A ZigBee network address is 16-bit long, so potentially 65,535 addresses can be assigned to all ZigBee devices in the tree (address 0 is reserved for the ZC). This amount should suffice for most applications.

A tree-based ZigBee network is characterized by *topological parameter*, which limits the height of the tree and the maximal number of children devices/ZRs that a ZC/ZR can have. By setting the topological parameter appropriately, we can roughly control the shape and extent of the tree. However, the actual topology also depends on the geographical distribution of devices.

Distributed Address Assignment Mechanism (DAAM) is the addressing method recommended by ZigBee specification. DAAM statically couples addresses with node locations in the tree. In DAAM, every possible location in the tree is pre-allocated an address that is unique yet routable. A ZC/ZR can locally allocate an address to its child with the knowledge of its own depth value and the global setting of the topological parameter. This addressing method has the merit that it simplifies the task of routing. When a ZC/ZR receives a packet not destined for it, the next-hop node can be deduced directly from the destination address without consulting a routing table. This eliminates the need for extra storage in ZC/ZRs to keep routing information.

Binding addressing with the topological parameter, however, causes the problem of unaddressing devices. When the setting of the topological parameter does not match the geographical distribution of devices, there may exist devices that are not addressable with DAAM while many addresses are still left unused. Unfortunately, there has been no automatic way to yield a matching parameter setting. So the problem is inherent.

We also remark *routing detour problem* associated with DAAM. When the path from some device to the ZR along the tree is not the shortest (in terms of hop count) among all potential ones, extra transmission is demanded by that path, which wastes precious bandwidth and energy resource.

In this paper, we propose three alternatives to alleviate the unaddressing and routing detour problems. The first approach, Centralized Stateful Address Configuration (CSAC), adapts conventional stateful addressing method to ZigBee trees. As CSAC creates no static binding between addresses and node locations in the tree, every available address is assignable to any node and the unaddressing problem occurs only for address exhaustion. The weakness of CSAC is the need for additional storage in every ZC/ZR to keep a routing table. The second approach is a hybrid method that utilizes DAAM with priority and applies CSAC only when needed. This approach is called Hybrid Address Configuration (HAC). The third approach, Router-Based Address Configuration (RBAC), partitions address space into chunks and assigns one chunk to each ZR on demands. A ZR then allocates available addresses from its chunk to its child ZEDs.

2. Preliminaries

Since ZigBee is a particular type of mobile ad hoc networks (MANETs), let us start with MANET addressing protocols. MANETconf [7] treated the problem of dynamically allocating a unique address to each node as a distributed agreement problem and proposed adapting a distributed mutual exclusion algorithm to MANET address configuration. Some approach partitions address space (using binary split) among nodes so that each node can configure new node independently [6]. These schemes are designed to deal with problems like node failure, message loss, node mobility, network partitioning and merge, and multiple concurrent initiations of the protocol. The ultimate goal is to ensure address uniqueness in despite of these problems, as well as to minimize the amount of unaddressed nodes. Refer to [3] for a comprehensive survey of current development of address configuration in MANETs.

Conventionally, MANET addressing methods view routing as an independent issue. They are not designed to assist routing tasks except for the guarantee of address uniqueness. In contrast, DAAM in ZigBee networks provides not only unique identification to every device, but also adequate routing information for every possible packet delivery path. The last property eliminates the need for an independent routing protocol.

Address space in [4] is defined by an n-dimensional coordinate system. Each coordinate, expressed as an n-tuple of integers, is considered a logical address. Once configured, a parent node assigns to a child node joining the network an unallocated logical address such that addresses of these two nodes differ in only one element in the tuple: the element in the child's address is one larger than that in the parent's. In this manner, the number of children nodes that each node can have is limited by n. Each node should also inform neighboring nodes which addresses have already been assigned by it, as multiple nodes may be eligible to assign the same address. The issue with this approach is that it imposes an n-dimensional mesh structure on the network, which is too complicated for some PAN applications. We focus on tree structure in the rest of this paper.

```
/* $d(P)$ denotes the depth value of $P$ */
/* $A(P)$ denotes the network address of $P$ */
/* $N_r(P)$ denotes the number of $P$'s children that are ZRs */
/* $N_c(P)$ denotes the number of $P$'s children that are ZEDs */
/* $\mathit{Cskip}(d) = \begin{cases} 1 + \mathrm{Cm} \times (\mathrm{Lm} - d - 1) & \text{if } \mathrm{Rm} = 1, \\ \frac{1+\mathrm{Cm}-\mathrm{Rm}-\mathrm{Cm}\times\mathrm{Rm}^{\mathrm{Lm}-d-1}}{1-\mathrm{Rm}} & \text{otherwise.} \end{cases}$ */

if $D$ is an FFD and $d(P) < \mathrm{Lm} - 1$ and $N_r(P) < \mathrm{Rm}$ then
    $N_r(P) \leftarrow N_r(P) + 1$    // accommodating $D$ as a ZR
    allocate $D$ the following address
        $A(P) + \mathit{Cskip}(d(P)) \times (N_r(P) - 1) + 1$
else if $d(P) \leq \mathrm{Lm} - 1$ and $N_c(P) < \mathrm{Cm}$ then
    $N_c(P) \leftarrow N_c(P) + 1$    // accommodating $D$ as a ZED
    allocate $D$ the following address
        $A(P) + \mathit{Cskip}(d(P)) \times \mathrm{Rm} + N_c(P)$
else
    // $D$ cannot be accommodated
end if
```

Figure 1. DAAM: The procedure for ZC/ZR P to allocate an address to a device D

The topological parameter associated with DAAM is a collection of three integer variables:

- Lm: the maximum depth value of the tree.
- Cm: the maximum number of children of a ZC/ZR.
- Rm: the maximum number of children of a ZC/ZR that can be ZRs.

According to ZigBee specification, the ZC is at depth 0 and devices at depth Lm can only be ZEDs, not ZRs. Although Lm, Cm, and Rm are all ranged from 0 to 14, their values are not independent as some value combinations are meaningless and some others require more addresses than allowed.

When operating in beacon-enabled mode, a ZC/ZR periodically broadcasts beacon frames to announce its presence and disclose related information. Any device should first scan for beacons before joining a PAN. The collected beacon information is used to build a neighbor table. The device then sends Association Request frames to a ZC/ZR that has the minimal depth value in the neighbor table. If the request is granted, an Association Response frame containing an allocated address (called *short address*) is sent back to the device requesting association. The procedure for a ZC/ZR to allocate an address to a device requesting association is detailed in Fig. 1. Note that when an FFD can be accommodated as a ZED if there is no room for ZRs in the ZC/ZR.

DAAM's addressing is hierarchical in the sense that any subtree possesses a block of consecutive addresses. Let P be a ZC/ZR located at depth d and D be a child of P. If D is a ZR, the subtree rooted at D is allocated $\mathit{Cskip}(d)$ sequential addresses, where $\mathit{Cskip}(d)$ is defined as [2]

$$\mathit{Cskip}(d) = \begin{cases} 1 + \mathrm{Cm} \times (\mathrm{Lm} - d - 1) & \text{if } \mathrm{Rm} = 1, \\ \frac{1+\mathrm{Cm}-\mathrm{Rm}-\mathrm{Cm}\times\mathrm{Rm}^{\mathrm{Lm}-d-1}}{1-\mathrm{Rm}} & \text{otherwise.} \end{cases}$$

With DAAM's hierarchical addressing, routing can be performed without consulting any routing table. Suppose a ZC/ZR at depth d with address A receives a packet destined for address $D \neq A$. If $A < D < A + \mathit{Cskip}(d-1)$, this packet is for some node in the subtree rooted at A and should be passed to the child with address $A + 1 + \lfloor (D - A - 1)/\mathit{Cskip}(d) \rfloor \times \mathit{Cskip}(d)$. Otherwise, the packet should be passed to A's parent. [5]

Therefore, the tree structure used by DAAM for address configuration also serves the purpose of routing. The routing path between any two devices is along the tree. This design is justifiable if ZigBee is to implement wireless sensor networks (WSNs). In a WSN, network traffic mostly flows into or comes from a specific node called sink, which is typically the ZC if the WSN is implemented with ZigBee. In contrast, network traffic in MANETs is conventionally assumed peer-to-peer, calling for MANET routing protocols that suit best for that traffic type.

A major weakness associated with DAAM is the lack of flexibility. The highest address that can be allocated by DAAM is $\mathit{Cskip}(0) \times \mathrm{Rm} + \mathrm{Cm} - \mathrm{Rm}$. So some addresses will be wasted if the topological parameter is not set appropriately. Also, as address space is statically partitioned among subtrees, it may occur that some subtree contains no more address to allocate while some others have plenty. The problem with DAAM is closely related to the following two protocol properties:

P1: The possibility that FFDs are accommodated as ZEDs.

P2: The limitation on the amount of devices allowed to associate with a ZC/ZR.

P1 holds when $\mathrm{Cm} > \mathrm{Rm}$ and P2 is intrinsic to DAAM.

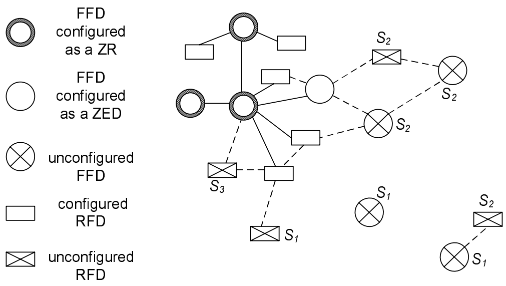

Figure 2. Examples of unconfigured devices. Solid lines stand for established associations while dashed lines are potential links.

A device may fail to acquire its address due to address shortage, i.e., the amount of usable addresses does not suffice for all devices in the network. Even if the possibility of address shortage is precluded, addressing failure may still occur to a device for one of the following causes:

- The device cannot reach any FFD within the communication range.
- The device can reach some FFDs, but none of them are ZRs.
- The device can reach some ZRs, but none of them is able to allocate an address to this device.

Let us define the following device sets to ease subsequent discussion:

- *UnC*: the set of unconfigured devices.
- *ZED*: the set of ZEDs.
- *ZR*: the set of ZRs.
- ZR_e: the set of ZR that are still able to allocate addresses to devices requesting associations.
- $N_F(d)$: For any device d, the set of FFDs that are within d's communication range.

Accordingly, the set *UnC* can be partitioned into three subsets:

- $S_1 = \{d \mid N_F(d) = \emptyset\}$.
- $S_2 = \{d \mid N_F(d) \neq \emptyset \wedge N_F(d) \cap \mathit{ZR} = \emptyset\}$.
- $S_3 = \{d \mid N_F(d) \cap \mathit{ZR} \neq \emptyset \wedge N_F(d) \cap \mathit{ZR}_e = \emptyset\}$.

Figure 2 illustrates some instances of these sets. In the following, we preclude the possibility of address shortage and discuss the source of each subset.

The size of S_1 depends on the density of FFDs, a factor irrelevant to addressing protocol design. In contrast, S_2 and S_3 are specific to the aforementioned protocol properties. P2 is the only reason why $\mathit{ZR} \neq \mathit{ZR}_e$ and therefore

Table 1. Possible sources of unconfigured device sets

	FFD density	P1	P2
S_1	Yes	No	No
S_2	No	Yes	No
S_3	No	No	Yes

the only source of S_3. For each device $d \in S_2$ and each $e \in N_F(d)$, we have either $e \in \mathit{ZED}$ or $e \in \mathit{UnC}$. The condition $e \in \mathit{ZED}$ is due to property P1. The other condition $e \in \mathit{UnC}$ is equivalent to $e \in S_1 \cup S_2 \cup S_3$, implying that any possible cause of addressing failure is also an indirect source of S_2. Table 1 summarizes possible sources of unconfigured device sets (S_1, S_2, and S_3), excluding the indirect source of S_2.

The routing detour problem also relates to the aforementioned protocol properties. To explain, we define a *potential path* for a device d to be a series of potential links between FFDs,

$$(d, f_1), (f_1, f_2), \cdots (f_{n-1}, f_n),$$

where $F = \{f_1, f_2, \ldots, f_{n-1}\}$ is a set of FFDs and f_n is the ZC. Such a potential path is not necessarily the routing path from d to the ZC; it becomes that only if (1) all the involved FFDs are ZRs and (2) each potential link ends up with an association. The routing detour problem occurs to d when neither of d's shortest potential paths is able to serve as a routing path from d to the ZC. For each such path, the following condition holds when d requests association with the network

$$\exists f_i \in F : f_i \in \mathit{ZED} \vee f_i \in \mathit{UnC} \vee (f_i \in \mathit{ZR} \wedge f_i \notin \mathit{ZR}_e).$$

The source of the condition $f_i \in \mathit{ZED} \vee f_i \in \mathit{UnC}$ is exactly that of S_2 while $f_i \in \mathit{ZR} \wedge f_i \notin \mathit{ZR}_e$ comes from the same reason why S_3 exists.

3. Flexible Address Configuration

The goal of this research is to devise alternative addressing methods for ZigBee networks. These methods are required to guarantee address uniqueness and, as an integrated part of the addressing scheme, form a tree structure for routing as well while alleviating addressing failure and routing detour problems.

Our proposal assumes that ZC acts as an address configuration server (ACS), which manages an address pool for entire network. An ACS is functionally equivalent to and can be implemented as a conventional DHCP server. It assigns unallocated addresses to devices on an on-demand basis. Each ZC/ZR is required to have adequate storage space to keep its routing table.

3.1. Centralized Stateful Address Configuration (CSAC)

Since ZC is the ACS, any one-hop neighbor of the ZC can acquire an address directly from the ZC on its association without difficulty (assuming no address shortage). A device that does not have a direct link to the ZC, however, cannot obtain an address directly from the ZC. When such a device attempts to associate with some ZR, the ZR should request an address from the ZC on behalf of the device. CSAC introduces two message types for this purpose: Address Request and Address Response. Detailed procedure follows.

- A ZigBee device attempts association with the network by sending Association Request to a neighboring ZR.
- On receiving the request, the ZR becomes the *proxy ZR* of the device. The proxy ZR then sends an Address Request message to the ZC on behalf of the device requesting association.
- The request message is delivered hop-by-hop to the ZC (We will discuss how to realize such a upward routing shortly.)
- The ZC allocates an unused address from the address pool and sends it to the proxy ZR by responding with an Address Response message.
- The response is delivered hop-by-hop to the proxy ZR (The issue concerning this downward routing will be addressed later.)
- The proxy ZR extracts the address from Address Response and sends it to the device attempting association by replying an Association Response.

Note this procedure is compliant to ZigBee's association procedure from the end device's point of view. With CSAC, FFDs are all ZRs while RFDs are all ZEDs. In contrast, FFDs may be degraded to ZEDs with DAAM.

We shall now discuss how a tree-based routing can be achieved. This relies on the following three properties.

Property 1 *Each ZR/ZED keeps a routing entry for its parent, which serves for the ZR/ZED's default route.*

Property 2 *Each ZC/ZR keeps one routing entry for each of its descendants (i.e., a host-specific route for each descendant). This entry points to the right next-hop device on the unique path that connects the ZC/ZR with the corresponding descendant.*

Property 3 *Each device contains no other routing entry.*

It is not hard to see that these three properties together guarantee successful operations for all possible networking scenarios:

- A device can send packets to any of its ancestors with default routes.
- A device can send packets to any of its descendants with host-specific routes.
- A device can send packets to any other devices by first delivering them to the nearest common ancestor of the source and destination (with default routes), from which the packets are then delivered to the destination with host-specific routes.

CSAC ensures the first property by requiring each device to take its parent as the default gateway upon successful association. To retain the second property, each device is required to initiate a route update procedure after it has been configured with an address, say, A_d. The procedure is described as follows.

- The device sends a Route Update message destined for the ZC.
- When the ZC or any halfway ZR P receives Route Update from its child C, P first creates a host-specific route in its routing table for address A_d with next-hop address set to C's address. P then forwards the update message to its parent if P is not the ZC.

In this way, the path from the ZC to each associated device can be created. The procedure also implies that the size of routing table in a ZC/ZR is proportional to the number of associated devices residing in the subtree rooted at the ZC/ZR.

3.2. Hybrid Address Configuration (HAC)

HAC is designed to reduce storage cost incurred by CSAC while retaining flexibility to a certain degree. HAC uses DAAM by default, and invokes CSAC only for devices that cannot be configured through DAAM. As mentioned, only address 0 to *Cskip*$(0) \times \mathrm{Rm} + \mathrm{Cm} - \mathrm{Rm}$ will be used by DAAM. The rest can therefore be utilized by CSAC.

When a ZC/ZR P receives an association request from a device D, P uses the procedure shown in Fig. 3 to allocate an address to D. Note that once P has been configured through CSAC, it can only use CSAC to allocate an address for D. P in this case cannot use DAAM because P's address is not within the range pertaining to DAAM. So there are in fact two types of ZRs in HAC: one configured via DAAM (called *D-ZR*) limits the number of child nodes while the other configured via CSAC (called *C-ZR*) does not have such limitation.

```
/* $d(P)$ denotes the depth value of $P$ */
/* $N_r(P)$ denotes the number of $P$'s children that are ZRs */
/* $N_c(P)$ denotes the number of $P$'s children that are ZEDs */

if $P$ has been configured through CSAC then
    allocate $D$ an address by running CSAC
else if $D$ is an FFD and $d(P) < \mathrm{Lm} - 1$ and $N_r(P) < \mathrm{Rm}$ then
    $N_r(P) \leftarrow N_r(P) + 1$      // accommodating $D$ as a ZR
    allocate $D$ a ZR address as defined by DAAM
else if $d(P) \leq \mathrm{Lm} - 1$ and $N_c(P) < \mathrm{Cm}$ then
    $N_c(P) \leftarrow N_c(P) + 1$      // accommodating $D$ as a ZED
    allocate $D$ a ZED address as defined by DAAM
else
    allocate $D$ an address by running CSAC
end if
```

Figure 3. HAC: The procedure for ZC/ZR P to allocate an address to a device D

If a device is configured by means of CSAC, all its ancestors, C-ZR or D-ZR, should create associative host-specific routes for it as a result of executing the route update procedure. Consequently, the size of routing table in a ZC/ZR is proportional to the number of associated descendants that are configured via CSAC. This is the reason why DAAM is used with priority: such strategy reduces storage cost.

The aim to save storage is also reflected by the following design philosophy. When an FFD cannot be accommodated as a ZR via DAAM, we can (1) accommodate it as a ZED through DAAM or (2) accommodate it as a C-ZR using CSAC. In Fig. 3, the former treatment takes precedence over the latter. While an alternative design that attempts these two options in reverse order may increase the ratio of associated devices, the adopted design is more likely to reduce storage cost.

There is also some change in routing rules with HAC. C-ZRs follow the routing rule of CSAC. For D-ZRs, whether a packet should be handled by DAAM's or CSAC's rule depends on the destination address. If the destination address is within the scope of DAAM's address space, DAAM's rule applies. Otherwise, CSAC's routing rule is in effect.

3.3. Router-Based Address Configuration (RBAC)

One drawback of CSAC comes from the additional communication cost between proxy ZRs and the ACS for address allocations. This cost can be reduced if Proxy ZRs own some spare addresses so that they could locally grant address requests without communicating with the ACS. This idea motivates RBAC.

RBAC is similar to CSAC. It partitions the whole address space into fixed-size blocks. The size of each block is a power of two. When a proxy ZR receives an association request, the proxy ZR sends Address Request to the ACS if (and only if) the association request is issued by an FFD. When the ACS receives the address request, it allocates an address block instead of a single address to the proxy ZR. The proxy ZR then informs the FFD of the block. The first address in the block is for the FFD and the rest are spares. After the association is completed, the FFD becomes a proxy ZR and can locally allocate spare addresses to RFDs that request associations with it.

As all ZEDs associated with the same ZR share an address block and the block size is known to every ZR, only ZRs need to initiate the route update procedure after their associations. ZEDs need not perform the procedure. As a result, each ZC/ZR keeps routing records only for ZRs residing in the subtree rooted at it. In contrast, a ZC/ZR with CSAC needs store addresses of *all devices* (both ZRs and ZEDs) in the same subtree.

Since the block size is a power of two, we can define a bit mask that indicates which bit in the address field should agree for two addresses being in the same address block. This notion is exactly the same as subnet mask used in subnetting IP networks. With this bit mask, a ZC/ZR can easily determine whether the destination address of a received packet is a ZR or ZED. If the destination is a ZED, the ZC/ZR can also determine the address of the ZR with which the ZED associates. The packet can then be forwarded to the destination ZR by consulting routing tables.

3.4. Analyses of Addressing Failure and Routing Detour

We shall now analyze whether these three protocols suffer from addressing failure and routing detour problems. FFDs can only be ZRs with CSAC or RBAC, so these two protocols do not have property P1. Unfortunately, HAC inherits P1 from DAAM since it uses DAAM with priority. CSAC neither has property P2, since it does not limit the number of devices associated with a ZC/ZR. HAC does not inherit P2 from DAAM due to the introduction of CSAC as a remedy. P2 holds for RBAC since the address block size confines the number of ZEDs (but not ZRs) allowed to associate with a ZC/ZR. Table 2 summarizes protocol properties.

From Tables 1 and 2, possibilities of unconfigured device sets (S_1, S_2, and S_3) with each protocol can be obtained (Table 3). It turns out that both the addressing failure and routing detour problems are most serious with DAAM and least with CSAC. The behaviors of HAC and RBAC should be in-between. In particular, the routing detour problem may arise with either DAAM, HAC, or RBAC, but not with CSAC.

Table 2. Summary of Protocol Properties

	DAAM	CSAC	HAC	RBAC
Forcing FFDs to be ZEDs? (P1)	Yes	No	Yes	No
Limiting associable devices? (P2)	Yes	No	No	Yes

Table 3. Possibilities of unconfigured device sets (S_1, S_2, and S_3) with each protocol

	DAAM	CSAC	HAC	RBAC
S_1	Yes	Yes	Yes	Yes
S_2	Yes	No	Yes	No
S_3	Yes	No	No	Yes

4. Empirical Results

We conducted extended simulations to investigate the performance of proposed schemes. We assumed an 1000×1000 m^2 deployment field, within which 200 to 1000 ZigBee devices were uniformly deployed at random. An additional device acting as the ZC was placed at the center of the deployment field. All devices had a communication range of 100 m. The ratio of FFDs to RFDs was one to one. We considered three settings for the topological parameter, which respectively stand for *tall* (Cm=4, Rm=2, Lm=14), *regular* (Cm=12, Rm=4, Lm=7), and *flat* (Cm=14, Rm=8, Lm=5) trees. For HAC, the topology setting was fixed to (Cm=12, Rm=5, Lm=6) for a fixed address-space partition between DAAM and CSAC. The block size for RBAC was set to 8. Each parameter setting was repeated 100 times to obtain average results.

4.1. Percentage of configured devices

The first metric we measured is the percentage of devices that were successfully configured with addresses. Both CSAC and RBAC performed the best, followed by HAC and then DAAM. Fig. 4 shows the result.

Since S_1 is the only source of configuration failure with CSAC, the curve of CSAC here also indicates a relative amount of S_1 devices. We found that FFD density did not suffice for an 100% configuration ratio until over 400 devices were deployed (note only half of them were FFDs).

Besides S_1, only S_3 contributes to RBAC's addressing failures. Therefore, the performance gap between CSAC and RBAC can be regarded as the degree of S_3's impact on RBAC's performance. Since RBAC's performance is hardly

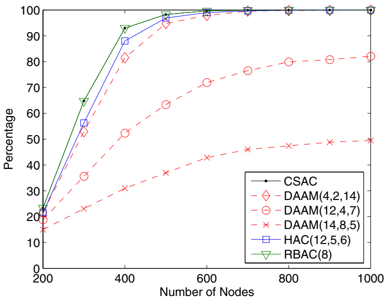

Figure 4. Percentage of devices being successfully configured

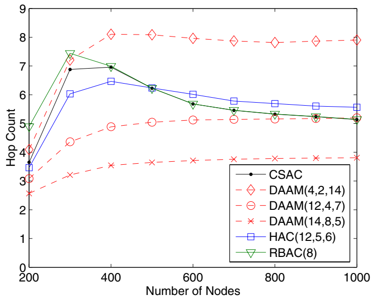

Figure 5. Average hop count from every device to the ZC

distinguishable from CSAC’s, we deduce that the impact is negligible.

DAAM’s failures were contributed by S_1, S_3, and S_3. S_1’s contribution diminished when over 400 devices were deployed. However, for DAAM with regular- and flat-tree settings, the configuration ratio only slightly increased with increasing device population. The significant performance gap between them and CSAC was mainly due to their relatively small settings on Lm. A small Lm will result in a large set of *FFD* $\cap$ *UnC* and hence a large set of S_2. This also explains why DAAM with the tall-tree setting had a better performance. The behavior of HAC was in-between, which is reasonable as HAC takes a hybrid design.

4.2. Hop Count

Hop count is a concern since proposed approaches place no limitation on tree depth, which may increase the path length between two potential packet-exchanging nodes. Fig. 5 shows the average hop count from every device to the ZC. The average hop count between each pair of nodes in the tree is similar to that given by Fig. 5 and is not shown here. The result indicates that DAAM with both regular- and flat- tree settings had relatively low hop-count values. However, the result with the tall-tree setting was the worst when 400 or more devices were deployed.

The superiority of DAAM with both regular- and flat-tree settings comes from its ability to confine tree depths. The difference between Figs. 4 and 5 reveals that DAAM is able to trade configuration ratios for hop counts by changing the value of Lm. A problem with that ability is the lack of automatic way to determine a suitable value for Lm that maximizes configuration ratio while minimizing depth value to the greatest possible extent.

CSAC and RBAC performed similarly. Their hop-count results arise initially with increasing nodes, but slightly reduce with more nodes. The reason for this trend is due to the following two competing factors:

- A tree formed by a few nodes is expected small in scale. So the average path length is also small. On the other hand, a large-scale tree may contain many nodes that have longer paths to the ZC, increasing the average value.

- When every device joins a tree, it seeks the shortest path (in terms of hop count) from it to the ZC and selects one of its neighbors that leads to this path as its parent. The selected path is *optimal* if the path length is exactly $\lceil d/r_t \rceil$, where d is the distance between the node and the ZC while r_t is the communication range. If node density is sufficiently high, devices are likely to find and select optimal paths. Otherwise, many nodes are forced to join the tree with sub-optimal paths connecting them.

When 300 or less nodes were deployed, the first factor dominated the results. But it was overtaken by the second factor when more than 400 devices were deployed. The behavior of HAS, again, demonstrates a tradeoff between DAAM and CSAC/RBAC.

4.3. Storage Cost

All proposed approaches demand storage to keep routing tables. We therefore take the size of routing tables as a gauge of storage cost. Fig. 6 displays average-case and worst-case storage costs associated with CSAC. The number of deployed devices was set to 400, 600, 800, and 1000. We can see that devices in upper levels generally had higher storage costs than those in lower levels. This is expected as one entry for each descendant is required in every ZC/ZR’s routing table. The most demanding nodes were those in depth one, where the table size in average was less than 33 while the maximal value can be as high as 300.

We took the results of CSAC as an evaluation basis, with which the results of HAC and RBAC were compared. We

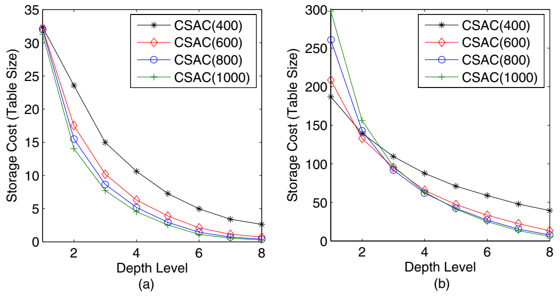

Figure 6. The size of routing tables in relation to depth values with CSAC. Numbers inside parentheses represent device population. (a) Average-case results (b) Worst-case results

define *improvement ratio* (IR) to be the ratio of cost reduction by the scheme in comparison to CSAC's cost. Fig. 7 shows IRs of HAC and RBAC in average and worst cases. The IR of RBAC was about 50% under all circumstances, meaning that RBAC halved CSAC's storage cost. In contrast, HAC's behavior was not stable. HAC was comparable to RBAC for devices with low depth values. However, its IR dropped as the depth value increased. The IR even became negative for devices with depth value six or higher. This is because in HAC, the size of routing table in a ZC/ZR is proportional to the number of descendants that are configured via CSAC. In out setting, ZRs with depth value five or lower were typically D-ZRs. This leads to positive IRs as descendants of D-ZRs were mostly associated using DAAM. In contrast, ZRs with depth value six or higher were all C-ZRs, meaning that DAAM can no longer help in storage-cost reduction. Furthermore, due to the routing detour problem, may devices were forced to associate with these C-ZRs. This increased the amount of their descendants and resulted in negative IRs.

Although HAC does not seem promising in IR comparisons, it still reduced the storage cost incurred by CSAC. In fact, the reduction is only slightly lower than that with RBAC. The reason for HAC's significant cost improvement despite of its unstable IRs is due to the fact that ZRs with low depth value dominate overall storage cost.

5. Conclusions

We have considered three new address configuration schemes accompanied with tree-based routing for ZigBee networks: CSAC, HAC, and RBAC. CSAC is the most flexible scheme such that it has achieved the highest percentage of configured devices in simulations. However, extra storage in every ZigBee router is required by CSAC, incurring a cost that is directly proportional to the number of associated descendants. HAC aims to reduce the storage cost while retaining flexibility to a certain degree. Its ability to make such a tradeoff has been demonstrated through simulations. RBAC's performance in terms of configuration ratio is hardly distinguishable from that of CSAC. Since RBAC also halves CSAC's storage cost, it is recommended as a remedy for the inflexibility problem brought about by DAAM, the conventional ZigBee address configuration scheme.

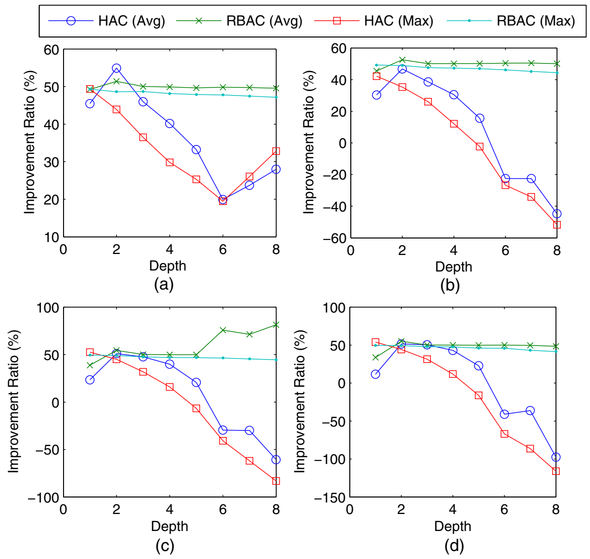

Figure 7. Average- and worst-case improvement ratios with HAC and RBAC. The number of deployed devices is (a) 400 (b) 600 (c) 800 (d) 1000.

References

[1] Part 15.4: Wireless medium access control (MAC) and physical layer (PHY) specifications for low-rate wireless personal area networks (LR-WPANs). IEEE Std. 802.15.4, 2003.

[2] ZigBee specification. ZigBee Alliance, Dec. 2006.

[3] C. Bernardos, M. Calderon, and H. Moustafa. Survey of IP address autoconfiguration mechanisms for MANETs. *IETF Internet Draft*, Oct. 2007.

[4] G. Bhatti and G. Yue. Structured addressing scheme for wireless networks, US Patent 0060134, Mar. 2007.

[5] G. Ding, Z. Sahinoglu, P. Orlik, J. Zhang, and B. Bhargava. Tree-based data broadcast in IEEE 802.15.4 and ZigBee networks. *IEEE Trans. on Mobile Computing*, 5(11):1561–1574, Nov. 2006.

[6] M. Mohsin and R. Prakash. IP address assignment in a mobile ad hoc network. In *Proc. MILCOM*, pages 856–861, 2002.

[7] S. Nesargi and R. Prakash. MANETconf: Configuration of hosts in a mobile ad hoc network. In *Proc. IEEE INFOCOM*, pages 1059–1068, 2002.