Evaluation of the Effects of a Grouping Algorithm on IEEE 802.15.4 Networks with Hidden Nodes

Jin-Yeong Um, Jong-Suk Ahn, and Kang-Woo Lee

Abstract: **This paper proposes hidden-node aware grouping (HAG) algorithm to enhance the performance of institute of electrical and electronics engineers (IEEE) 802.15.4 networks when they undergo either severe collisions or frequent interferences by hidden nodes. According to the degree of measured collisions and interferences, HAG algorithm dynamically transforms IEEE 802.15.4 protocol between a contention algorithm and a contention-limited one. As a way to reduce the degree of contentions, it organizes nodes into some number of groups and assigns each group an exclusive per-group time slot during which only its member nodes compete to grab the channel. To eliminate harmful disruptions by hidden nodes, especially, it identifies hidden nodes by analyzing the received signal powers that each node reports and then places them into distinct groups. For load balancing, finally it flexibly adapts each per-group time according to the periodic average collision rate of each group.**

This paper also extends a conventional Markov chain model of IEEE 802.15.4 by including the deferment technique and a traffic source to more accurately evaluate the throughput of HAG algorithm under both saturated and unsaturated environments. This mathematical model and corresponding simulations predict with 6% discrepancy that HAG algorithm can improve the performance of the legacy IEEE 802.15.4 protocol, for example, even by 95% in a network that contains two hidden nodes, resulting in creation of three groups.

Index Terms: **Analytical models, grouping algorithms, hidden nodes (HN), IEEE 802.15.4, sensor networks.**

I. INTRODUCTION

Institute of electrical and electronics engineers (IEEE) 802.15.4 protocol has been introduced with an aim of being employed as a medium access control (MAC) protocol of low rate wireless personal area networks (LR-WPANs) which are supposed to operate for a long period of time without consuming much energy. To meet various applications' demands, it is also designed to provide both synchronous and asynchronous services for which it divides its superframes into two periods such as contention free period (CFP) and contention access period (CAP), respectively. To guarantee timely transmission without any delay due to contention, CFP runs a static time division multiple access (TDMA) technique whereas CAP adopts a carrier sense multiple access with collision avoidance (CSMA/CA) algorithm to maximize the utilization of a single shared wireless broadcast channel [1], [2].

The performance of CAP has been well known to be rapidly deteriorated under two situations. The first case corresponds to overcrowded networks where a large number of nodes contend to send their data. When more than 8 nodes compete to capture the channel, for example, IEEE 802.15.4 wastes significant amount of bandwidth to resolve collisions, leading to only around 40% of utilization of the underlying physical link capacity [3]. In IEEE 802.15.4, furthermore, collisions tend to occur more often than in IEEE 802.11 due to the lack of the freezing operation and the small size of maximum backoff window. Note that once the channel is polled to be busy, IEEE 802.11 stops its backoff timer from ticking off until the channel becomes free. The absence of this freezing operation forces more nodes of IEEE 802.15.4 to be densely packed into the last backoff stage, causing more collisions. The small maximum backoff window size which is 128, amounting to one tenth of that of IEEE 802.11 is also likely to increase the collision probability as more nodes are involved in competition.

The second degrading case occurs under the networks populated with hidden nodes, often inflicting contaminations on outstanding frames. A hidden node is defined as the one which cannot sense its neighbor nodes' transmission signal due to either their limited radiation range or obstacles. Note that the number of hidden nodes tends to be in inverse proportion to the transmission power of nodes. As IEEE 802.15.4 tries to spend as little energy as possible, it is likely to more severely suffer from the hidden node problem. This problem has been confirmed to be detrimental to the performance of wireless networks. A paper [4] reports that throughput abruptly drops by up to 62% even when only one node is hidden in a 20-node network. Another paper [5] shows that the probability that any two nodes are hidden from each other reaches up to 40% when nodes are randomly distributed within a certain communication range.

Under the presence of hidden nodes and heavily competitive environments, request-to-send (RTS) and clear-to-send (CTS) frames provide an effective mechanism to prevent the performance from steeply falling down [6], [7]. This RTS/CTS scheme not only shortens collision duration but also lets the status of receivers visible to hidden nodes. It, however, is rarely employed in energy-sensitive networks like IEEE 802.15.4 since it always requires two short frames to be exchanged before every data transmission, leading to aggravating energy consumption and incurring some extra delay per frame transmission. IEEE 802.15.4, furthermore, gains no benefit of less collision duration since the size of data frames is not much larger than that of RTS/CTS frames.

Manuscript received January 11, 2013; approved for publication by Yousefi'zadeh, Homayoun, Division II Editor, November 28, 2013.

This research is supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (No. 2012-2020051) and the research program of Dongguk University, 2013.

The authors are with Computer Engineering Department, Dongguk University, 26, Pil-dong 3-ga, Jung-gu, Seoul, Korea, email: {mog07, jahn, klee} @dongguk.edu.

Digital object identifier 10.1109/JCN.2014.000011

To improve the performance of IEEE 802.15.4 especially when it suffers from heavy congestion and interferences, this paper proposes a grouping algorithm named as hidden-node aware grouping (HAG) which tries to control the degree of contentions without losing the efficiency of CSMA/CA. According to the measured congestion level, HAG algorithm dynamically converts IEEE 802.15.4 to a contention-limited algorithm by maintaining some number of groups into which it evenly distributes covered nodes and isolates hidden nodes. Each group is assigned its own transmission interval during which only its own member nodes are allowed to grapple for the channel.

In detail, HAG algorithm run on the coordinator of IEEE 802.15.4 consists of three steps; measurement of collision rates (MCR), collection of received signal power (CRSP), and regrouping of nodes (RN). The MCR step evaluates the average congestion rate of each existing group which combines both the collision rate and the interference rate. When the average congestion rate of a group exceeds a predetermined upper threshold or hidden nodes are detected to be existent, HAG algorithm calls CRSP procedure after MCR step while otherwise it directly jumps to RN procedure. CRSP locates hidden nodes by scanning the received signal power tables reported from all the nodes in the group. Note that each node in HAG algorithm records the power of received signals from its senders in a table named as the received signal power table.

If hidden nodes are recognized in a given group, RN procedure creates the same number of groups as the number of hidden nodes and then separates them into distinct groups. After this isolation, it tries to evenly place the remaining nodes into the groups with an aim of balancing each group size. If there is no hidden node, however, it just generates an additional new group and redistributes the nodes into the existing groups including the new one.

When the average congestion rates of some groups are below a predetermined lower threshold, it merges two groups with the lowest and second lowest average rates. After completing this grouping process, HAG algorithm assigns the initial group time to the newly created groups and the adjusted group time based on the congestion rates to the existing groups. Finally, HAG algorithm broadcasts an extended beacon message to inform all nodes of their group assignment and per-group time, namely the time interval allocated to a group. After this broadcast, each node runs CSMA/CA algorithm to acquire the channel during its group time.

For accurate evaluation of HAG algorithm's performance under saturated and unsaturated conditions with hidden nodes, this paper supplements the legacy performance model of IEEE 802.15.4 with the deferment algorithm [8], [9] and a traffic generator. Note that the deferment algorithm postpones the transmission of a frame to the next superframe if the transmission of a frame cannot be completed within the current superframe due to its deficient remaining time. It is crucial to be included in the model since it heavily affects the performance as the ratio of the frame size to the superframe size becomes comparable. In HAG algorithm, frames are more probable to be transferred during the next group time when the per-group time becomes shorter.

According to our proposed analytical model and corresponding NS-2 simulations, the deferment algorithm can severely deteriorate the performance of IEEE 802.15.4 by 30% especially when the ratio of the frame size to the superframe duration becomes larger. These two evaluation tools, furthermore, forecast that HAG algorithm improves the throughputs of IEEE 802.15.4 by 87% and 95% in comparison to RTS/CTS mechanism when IEEE 802.15.4 networks contain one and two hidden nodes respectively, leading to creation of two and three groups.

This paper is organized as follows. Section II reviews related work and Section III explains the extended analytical model covering the deferment algorithm and a traffic source for precisely analyzing IEEE 802.15.4 networks with hidden nodes. Section IV elaborates HAG algorithm in detail and Section V presents the mathematical throughput model for HAG algorithm. Section VI lists the results of conducted experiments to verify the accuracy of the analytical model and provides the degree of improvements compared to the legacy IEEE 802.15.4. Lastly, Section VII summarizes conclusions and future research issues.

II. RELATED WORK

This section explains two typical approaches to reduce the impact of hidden nodes on the throughput of wireless networks. The first approach corresponds to a virtual channel sensing technique to stretch the physical channel sensing range to cover the entire network with the help of RTS/CTS control frames. These control frames explicitly let every node monitor the status of receivers before it starts to send full-size data frames. Since the size of these control frames is smaller than that of data frames, they dampen the harmful effect of hidden nodes by shortening collision duration [6], [7]. However, they waste some amount of bandwidth since they should be always sent ahead of data frames regardless of the existence of hidden nodes. Since they also cannot solve the exposed station problem due to that the following acknowledgment (ACK) frames fed back to the sender are interfered, they are rarely employed in real wireless networks [10].

The second category is grouping algorithms which differentiate transmissions from hidden nodes by isolating hidden nodes into different groups which are assigned non-overlapped time interval. One technique of this approach is to exploit global positioning system (GPS) service to recognize the locations of hidden nodes [11]. Based on each node's positional information, the coordinator divides nodes into separate groups. Since this algorithm requires GPS service which is expensive and provides limited distance resolution, it is not suitable for IEEE 802.15.4 networks which target to operate in small area with low cost.

Another technique [5] is to decide the existence of hidden nodes using a polling procedure in which the coordinator sends a poll message to every node and each polled node responds with poll-acknowledgement. After a polling procedure, each node orderly reports a list of nodes whose acknowledgements it has heard. Based on these reports, the coordinator establishes the hidden relations and disjoints hidden nodes into different groups. For better throughput, this scheme also allocates per-group time to each group in proportion to the number of its member nodes.

Among the conventional grouping algorithms, HAG algorithm closely resembles the one presented in [5]. They both, for example, group nodes with an aim to allocate hidden nodes into different groups. They, however, differ in two aspects. First, our technique focuses on comparing the performance of a grouping algorithm with that of IEEE 802.15.4 at the presence of hidden nodes whereas [5] emphasizes on calculating the maximum number of groups required for disjointing hidden nodes. To increase accuracy of the analytical model, furthermore, our model contains the deferment algorithm and a traffic source that were not considered in [5]. Second, HAG algorithm adopts different techniques in every step ranging from detecting hidden nodes to allocating per-group intervals. To infer hidden relations, for instance, it uses a signal power table constantly filled by each node during its normal operation time whereas [5] invokes a lengthy procedure for poll-and-response and then gathers the list of recognizable nodes from every node. It, finally, dynamically readjusts the per-group time depending on average congestion rates.

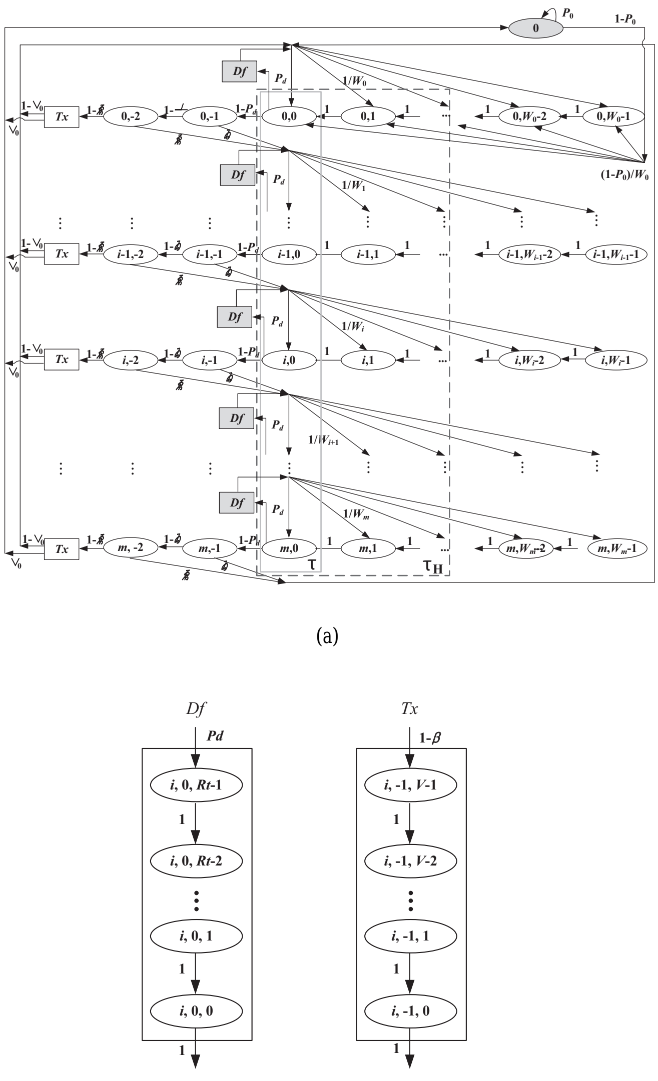
III. EXTENDED ANALYTICAL MODEL FOR IEEE 802.15.4 WITH HIDDEN NODES

This section explains an extended 2-dimensional Markov chain model, named as extended analytical model (EAM) for IEEE 802.15.4 networks with hidden nodes [4], [12]–[14] which augments the deferment technique to heighten the accuracy. EAM also includes a traffic source of Poisson process to enable the prediction of IEEE 802.15.4 behaviors under unsaturated environments.

The entire view of EAM is depicted in Fig. 1 where these two added features are highlighted by the shaded boxes labeled with Df and the shaded top circle marked with 0, respectively. The state of IEEE 802.15.4 at a given time is represented by circles in Fig. 1(a) while numbers on arcs indicate the transition probability from one state to its neighbor's state. Each circle contains two variables (i, k), standing for current backoff stage and remaining backoff timeout. When k becomes either -1 or -2, these states represent the first and second clear channel access (CCA) operations. Note that IEEE 802.15.4 performs two channel sensing operations named as CCA after the backoff timeout is expired. Fig. 1(a) assumes that for a given frame IEEE 802.15.4 goes through m maximum backoff trials accounted by the bottom row of circles and then resets its backoff timer.

Figs. 1(b) and 1(c) present the magnified view of the two small boxes labeled with Df and Tx in Fig. 1(a), respectively. One additional state variable t out of (i, k, t) in Df and Tx corresponds to the remaining superframe time and the remaining time before completion of a frame's transmission, respectively. Note that Tx box models a frame's transmission delay V. P_d over arcs toward Df box denotes the transition probability that the remaining time within the current superframe is insufficient so that the transmission of the frame should be delayed to the next superframe. For derivation of P_d, please refer to [9].

The box with broken lines labeled by τ_H in Fig. 1(a) denotes the vulnerable area which may cause the clash of the currently ongoing transmission when hidden nodes fall into one of states in this area. τ stands for the probability of carrying out the first CCA which is equal to the sum of all the probabilities of the states contained in the inner box labeled with τ in Fig. 1(a). Note that the width of τ_H is equal to the transmission delay of a data frame and its acknowledgement frame plus short interframe space (SIFS) under the assumption that the sizes of data and acknowledgement frames are fixed.



(a)

(b) (c)

Fig. 1: Markov chain model: (a) 2-dimensional Markov chain of IEEE 802.15.4 with deferment algorithm and traffic source, (b) magnified view of deferment box Df, and (c) magnified view of transmission box Tx.

Once IEEE 802.15.4 manages to successfully send a frame, it goes back to the uppermost circle of Fig. 1(a) marked by 0 from which it determines the availability of a new frame. From this state, it runs into the backoff stage 0 with the probability $(1-P_0)$ that the upper layer passes down a new frame during the average time slot, $E[slot]$. Otherwise it stays at this state with the probability P_0. Since we assume that new data are produced by a Poisson process with an average rate of λ, P_0 is calculated as $P_0 = e^{-\lambda E[slot]}$.

Table 1 succinctly summarizes descriptions of the important parameters and Table 2 compares closed-form equations of eight performance metrics derived from conventional analytical model (CAM) and EAM [4]. Table 2 also displays how per-

Table 1: Description of performance important parameters.

Symbol	Description
$b_{i,j}$	Probability that the system backs off i times and the backoff timeout is j time slots
τ	Probability that a node carries out its first CCA operation
τ_H	Probability that a node performs its first CCA even though the underlying channel is occupied by hidden nodes
α, α_H	Probability that a node backs off its backoff timer at the first CCA operation ($\alpha_H = \alpha$ in the presence of hidden nodes)
β, β_H	Probability that a node backs offs its backoff timer at the second CCA operation ($\beta_H = \beta$ in the presence of hidden nodes)
S, S_H	Throughput of IEEE 802.15.4 networks ($S_H = S$ in the presence of hidden nodes)
p	Probability that a node retreats to the next below backoff stage during two CCAs, namely $p = \alpha + (1 - \alpha)\beta$
V	Transmission delay of one frame
X	Minimum number of backoff stages whose timeout is larger than V, the transmission delay of one frame
n	Total number of nodes, $n = n_C + n_H$ where n_C is the number of covered nodes to recognize the presence of nodes and n_H is the number of hidden nodes
P_{tr}	Probability that at least one node transmits one frame
P_S	Probability that a given node successfully sends one frame
T_S	Time spent to successfully send one frame
T_C	Time taken to finish frame collision
L_{pl}	Length of user data excluding the frame header
R_t	Average remaining time when the transmission of a frame should be deferred

formance equations are changed when they account for hidden nodes. For example, α_H denotes the same probability as α except that it influences the effect of hidden nodes. In Table 2, $b_{0,0}$ is the state probability that IEEE 802.15.4 stays at state $(0, 0)$ in Fig. 1(a). The expression of $b_{0,0}$ is derived from two equations, one for expressing a given state probability $b_{i,j}$ in terms of $b_{0,0}$ and the other for identifying the summation of all state probabilities with 1. For derivations of other performance metrics, please refer to [4].

To evaluate the effect of hidden nodes on the performance of IEEE 802.15.4 networks, at first Fig. 2 plots three pairs of throughput graphs forecast by CAM and EAM as a function of the number of nodes while varying the number of hidden nodes (HN). Note that the throughput in Fig. 2 represents the ratio of the effective speed predicted by each model to the physical channel capacity. For this analysis, we assume that the duration of one superframe is 122.88 ms from *aBaseSuperframeDuration* * 2^{SO} where *aBaseSuperframeDuration* and *superframe order (SO)* are set to 15.36 ms and 3 while the sizes of data and acknowledgement frames are 70 bytes and 11 bytes, respectively [1], [2].

Fig. 2 shows the performance gap between EAM and CAM as a function of the number of covered nodes and hidden nodes. As shown in Fig. 2, EAM predicts lower throughput than CAM due to the deferment mechanism regardless of the number of nodes. Fig. 2 also describes that the ratio of these performance gaps between these two models becomes larger as HN increases. In sparse networks, for example, this throughput gap widens by up-to 80% or 95% when HN is either 1 or 3, respectively. These performance differences gradually disappear as the number of nodes increases, meaning that the effect of collisions overrides that of interferences by hidden nodes in densely populated networks.

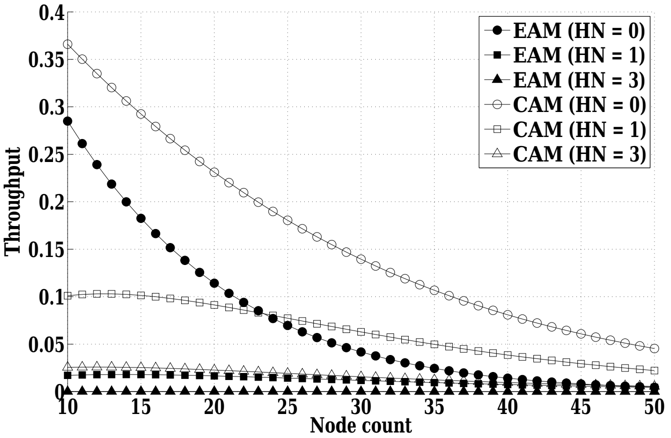


Fig. 2: Comparison of throughput predicted by CAM and EAM.

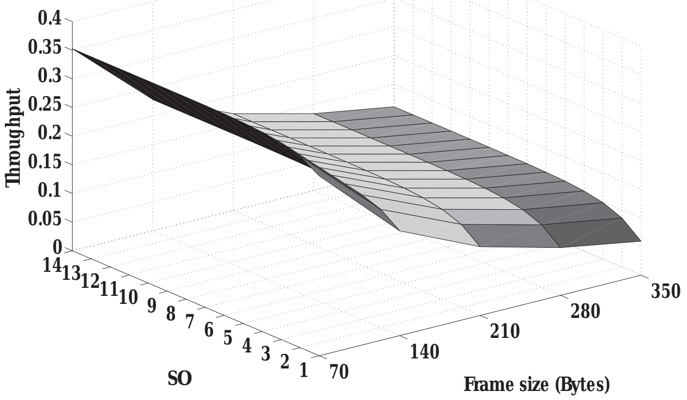


Fig. 3: Throughput variation as a function of superframe and data frame sizes.

Fig. 3 draws the throughput variations of EAM due to the deferment scheme as the sizes of data frames and superframe vary in 10-node network without hidden nodes. From Fig. 3, it is observed that as the ratio of the size of data frames to that of superframes becomes bigger, the throughput tends to be rapidly lowered, leading to 70% degradation compared to the maximum throughput when the size of data frames is one third of that of a superframe. This deterioration verifies that the analytical model of IEEE 802.15.4 should include the deferment algorithm for accurate evaluation of IEEE 802.15.4 performance especially when the frame size gets comparable to the superframe period.

Fig. 4 confirms that EAM with a Poisson traffic generator can accurately evaluate unsaturated IEEE 802.15.4 networks by comparing the results of EAM to those of simulations. Fig. 4 simulates a simple network consisting of one source and its destination. The comparison of two graphs in Fig. 4 proves that their deviations rarely exceed 6% maximally. For comparison purpose, Fig. 4 also displays throughputs of CAM which outperforms EAM due to the lack of the deferment algorithm.

Table 2: Comparison of analytical equations for important performance metrics.

Symbol	Equations from CAM	Equations from EAM
$b_{0,0}$	$\frac{2(1-p)(1-2p)}{\left[W(1-p)(2-(2p)^{m+1})+(1+2p)(5-2\alpha+2(1-p)V)(1-p^{m+1})\right]}$	$\frac{2(1-2p(1-P_d))(1-p(1-P_d))(1-P_0)}{\left[W((1-[2p(1-P_d)]^{m+1})(1-P_d)[1-p(1-P_d)](1-P_0)+(1-[p(1-P_d)]^{m+1}))(1-2p(1-P_d))[2\pi_0+(1-P_0)(1-P_d)(3-2(1-\alpha)+2V(1-p))+2pRt]\right]}$
τ	$b_{0,0}\frac{1-p^{m+1}}{1-p}$	$b_{0,0}\frac{1-[p(1-p_d)]^{m+1}}{1-[p(1-p_d)]}$
τ_H	$b_{0,0}\left[\left(\frac{1-p^X}{2(1-p)}+\frac{W(1-(2p)^X)}{2(1-2p)}\right)+\left((V+1)\frac{p^X-p^{m+1}}{1-p}-\frac{V(V+1)}{2W}\frac{(\frac{p}{2})^X-(\frac{p}{2})^{m+1}}{1-\frac{p}{2}}\right)\right]$	$b_{0,0}\left[\left(\frac{(1-P_d)(1-(p(1-P_d))^X)}{2(1-p(1-P_d))}+\frac{W(1-P_d)(1-(2p(1-P_d))^X)}{2(1-2p(1-P_d))}\right)+\left((V+1)(1-P_d)\frac{(p(1-P_d))^X-(p(1-P_d))^{m+1}}{1-p(1-P_d)}-\frac{V(V+1)(1-P_d)}{2W}\frac{(\frac{p(1-P_d)}{2})^X-(\frac{p(1-P_d)}{2})^{m+1}}{1-\frac{p(1-P_d)}{2}}\right)\right]$
α	$V[1-(1-\tau)^{n-1}](1-\alpha)(1-\beta)$	$V[1-(1-\tau)^{n-1}](1-p_d)(1-\alpha)(1-\beta)$
β	$\left[1-\left(\frac{P_{tr}}{P_{tr}(1+\frac{1}{1-(1-\tau)^n})}\right)\right](1-(1-\tau)^n)$	$\left[1-\left(\frac{P_{tr}}{P_{tr}(1+\frac{1}{1-(1-\tau)^n})}\right)\right](1-(1-\tau)^n)$
α_H	$V[1-(1-\tau)^{n_C-1}](1-\alpha)(1-\beta)$	$V[1-(1-\tau)^{n_C-1}](1-p_d)(1-\alpha)(1-\beta)$
β_H	$\left[1-\left(\frac{P_{tr}}{P_{tr}(1+\frac{1}{1-(1-\tau)^{n_C}})}\right)\right](1-(1-\tau)^{n_C})$	$\left[1-\left(\frac{P_{tr}}{P_{tr}(1+\frac{1}{1-(1-\tau)^{n_C}})}\right)\right](1-(1-\tau)^{n_C})$
S	$\frac{n\tau(1-\alpha)(1-\beta)P_S L_{pl}}{\left[(1-\tau)+\tau\alpha+2\tau(1-\alpha)+\tau(1-\alpha)(1-\beta)[P_S T_S+(1-P_S)T_C]\right]}$	$\frac{n\tau(1-\alpha)(1-\beta)P_S L_{pl}}{\left[(1-\tau)+(1-p_d)(\tau\alpha+2\tau(1-\alpha)+\tau(1-\alpha)(1-\beta))[P_S T_S+(1-P_S)T_C]\right]}$
S_H	$\frac{n\tau(1-\alpha_H)(1-\beta_H)P_S L_{pl}}{\left[(1-\tau)+\tau\alpha_H+2\tau(1-\alpha_H)+\tau(1-\alpha_H)(1-\beta_H)[P_S T_S+(1-P_S)T_C]\right]}$	$\frac{n\tau(1-\alpha_H)(1-\beta_H)P_S L_{pl}}{\left[(1-\tau)+(1-p_d)((\tau\alpha_H+2\tau(1-\alpha_H)+\tau(1-\alpha_H)(1-\beta_H))[P_S T_S+(1-P_S)T_C]\right]}$

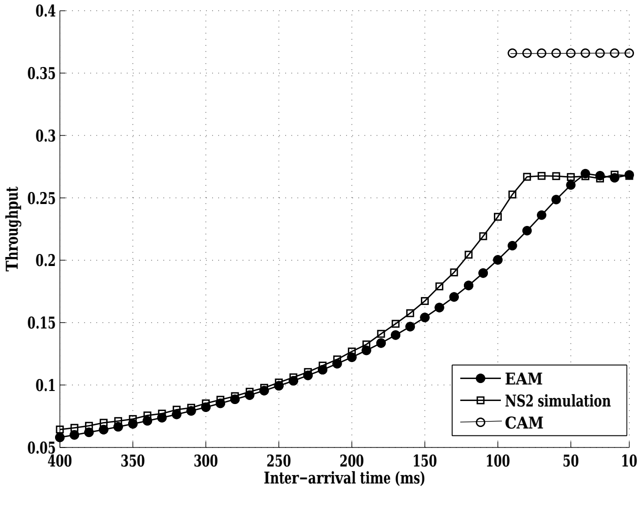


Fig. 4: Comparison of throughputs of EAM, CAM, and simulations under unsaturated conditions.

IV. HAG ALGORITHM

This section outlines HAG algorithm that is periodically run by the coordinator. The following three subsections elaborate three main steps of HAG algorithms whose pseudo codes are sketched in Fig. 5; MCR, CRSP, and RN. The forth one depicts the necessary modifications on the standard IEEE 802.15.4 frame header for accommodating HAG algorithm and a dynamic per-group time allocation algorithm for performance optimization. The last one addresses some implementation issues of HAG algorithm.

A. MCR

As shown in Fig. 5, to decide whether to regroup nodes, the coordinator at first calls *EstimateAvgColRate*() to calculate the average collision rate of every existing group at each time period called *HAG_PERIOD*. Note that HAG algorithm initially assigns all registered nodes into the default group numbered as 1. *EstimateAvgColRate*() estimates *AvgColRate*(n), the average collision rate of all groups as depicted in (1) where N and *AvgColRate*$_i(n)$ represent the total number of groups and the average collision rate experienced by group i at the measurement time n respectively. *AvgColRate*$_i(n)$ is iteratively computed as in (2) where *CurrColRate*$_i(n)$, *AvgColRate*$_i(n-1)$, and w_i correspond to the current collision rate, the average collision rate at the previous time, and the weight factor between 0 and 1, respectively. Note that the subscript i of each parameter in the following equations indicates the group identifier which ranges from 1 to *GroupNumber*, the number of cur-

```
1   HAG() {
2     AvgColRate(n) = EstimateAvgColRate();
3     if(IsAbove(AvgColRate(n), CONG_UPPER_THRESHOLD)){
4       if(HiddenNodeExist) {
5         SigPowerTable = BuildSigPowerTable();
6         GroupNumber = ReGroup(SigPowerTable, GroupTable, NULL);
7       } else
8         ReGroup(NULL, NULL, GroupNumber++);

9     else if(IsBelow(AvgColRate(n), CONG_LOWER_THRESHOLD)) {
10        GroupNumber--;
11        MergeGroup();
12    }}}
```

Fig. 5: Main pseudo code of HAG algorithm.

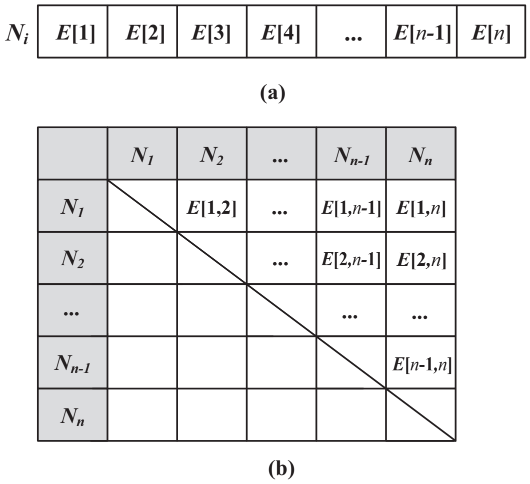
rently established groups. exponentially weighted moving average (EWMA) method is used as the smoother for the average collision rate.

$$AvgColRate(n) = \frac{\sum_{i=1}^{N} AvgColRate_i(n)}{N}, \qquad (1)$$

$$\begin{aligned} AvgColRate_i(n) &= w_i \times CurrColRate_i(n) \\ &\quad + (1 - w_i) \times AvgColRate_i(n-1). \qquad (2) \end{aligned}$$

For $CurrColRate_i(n)$, the coordinator divides the number of corrupt frames with the total number of transmitted frames in group i during *HAG_PERIOD*. For corrupt frames, it counts frames garbled due to either collision or interference with different weights whenever it receives them. HAG algorithm gives more weight to interference than to collision since hidden nodes are known to more severely deteriorate the performance. In our implementation, for example, one frame contaminated by interference is counted as 1.5 frames. When interference is detected, furthermore, HAG algorithm sets the interference flag, *HiddenNodeExist*. Note that $CurrColRate_i(n)$ becomes 0 when group i has not delivered any frames during *HAG_PERIOD*. Collisions are differentiated from interference depending on which part of frames are corrupt [15], [16]. If neither physical (PHY) protocol data unit (PPDU) header nor MAC protocol data unit (MPDU) is received successfully, then corruption is considered to happen due to collision. On the other hand if the coordinator recognizes PPDU header without any error but fails to accurately decode MPDU, it figures out that this failure is caused by hidden nodes.

After evaluating *AvgColRate*(n), the coordinator decides whether to regroup nodes by comparing *AvgColRate*(n) with *CONG_UPPER_THRESHOLD* as illustrated at line 3 in Fig. 5. If *AvgColRate*(n) is above the threshold and the *HiddenNodeExist* flag is set to *TRUE*, it performs *CollectSigPowerTable*() to pinpoint which nodes are hidden from others. After that, the coordinator calls *ReGroup*() to disperse hidden nodes into different groups. Otherwise, when the network is either overcrowded or underutilized without hidden nodes, it adjusts the number of groups by invoking *ReGroup*() to divide the most congested group into two or *MergeGroup*() to merge the least and the second least crowded group into one.



N_i	E[1]	E[2]	E[3]	E[4]	...	E[n-1]	E[n]

(a)

	N_1	N_2	...	N_{n-1}	N_n
N_1		E[1,2]	...	E[1,n-1]	E[1,n]
N_2			...	E[2,n-1]	E[2,n]
...				...	...
N_{n-1}					E[n-1,n]
N_n					

(b)

Fig. 6: Signal power table composed by coordinator: (a) Signal power vector of node i and (b) signal power table $E[\cdot]$.

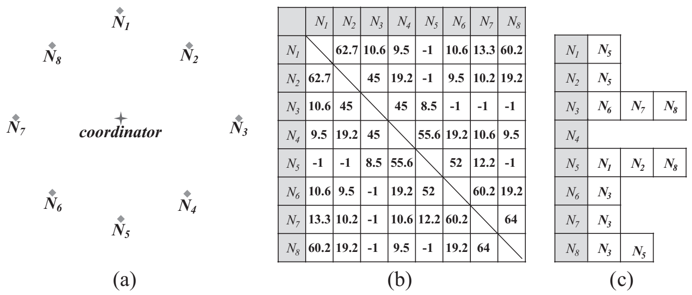
B. CRSP

When *AvgColRate*(n) is greater than *CONG_UPPER_ THRESHOLD* and *HiddenNodeExist* is *TRUE*, HAG algorithm calls *CollectSigPowerTable*() to identify hidden nodes and then performs *ReGroup*() to separate them into different groups. *CollectSigPowerTable*() composes *SigPowerTable* using signal power vectors informed from each node whose format is drawn in Fig. 6(a). Note that in IEEE 802.15.4 each node constantly evaluates other nodes' transmission energy through *PLME-ED*() application programming interface (API) of IEEE 802.15.4 physical layer [1], [2]. IEEE 802.15.4 standard specifies that each node should measure the signal strength and make this data available to the upper layer.

Each element $E[j]$ conveyed by N_i namely node i in Fig. 6(a), for example, represents the strength of signal delivered from N_j. Note that the greater $E[j]$, the closer N_j is to N_i. When $E[j]$ is set to the initial value -1, it indicates that either N_j has not sent any frame up to that time or N_j is hidden from N_i. After collecting all nodes' vectors, the coordinator builds a signal power table as presented in Fig. 6(b). Note that only the upper triangle of this table contains some power strengths under the assumption that $E[i,j]$ equals to $E[j,i]$. If the wireless link between N_i and N_j is asymmetric, however, HAG algorithm takes the average of the two received powers, $E[i,j]$ and $E[j,i]$.

C. RN

Once the signal power table $E[\cdot]$ is completely filled up, the coordinator executes *ReGroup*() which consists of three subprocedures, *DrawHiddenGraph*(), *BuildHiddenGroup*(), and *CompleteGroup*() for establishing a table to put hidden nodes in a row, distributing hidden nodes into separate groups, and placing other remaining nodes into the groups, respectively. The first step, *DrawHiddenGraph*() constructs the hidden node relation table $H[\cdot]$ where the ith row is comprised of a list of nodes



	N_1	N_2	N_3	N_4	N_5	N_6	N_7	N_8
N_1		62.7	10.6	9.5	-1	10.6	13.3	60.2
N_2	62.7		45	19.2	-1	9.5	10.2	19.2
N_3	10.6	45		45	8.5	-1	-1	-1
N_4	9.5	19.2	45		55.6	19.2	10.6	9.5
N_5	-1	-1	8.5	55.6		52	12.2	-1
N_6	10.6	9.5	-1	19.2	52		60.2	19.2
N_7	13.3	10.2	-1	10.6	12.2	60.2		64
N_8	60.2	19.2	-1	9.5	-1	19.2	64	

(b)

N_1	N_5		
N_2	N_5		
N_3	N_6	N_7	N_8
N_4			
N_5	N_1	N_2	N_8
N_6	N_3		
N_7	N_3		
N_8	N_3	N_5	

(c)

Fig. 7: An example process for establishing hidden node relation table.

hidden from node N_i.

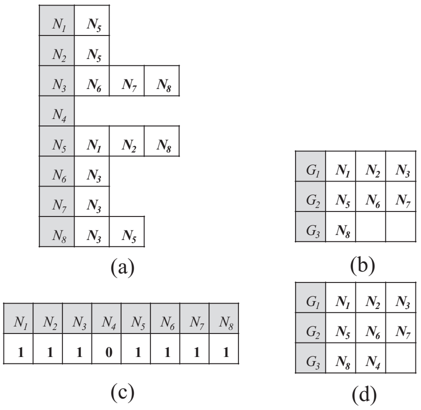
Fig. 7 illustrates an exemplary process establishing a hidden node relation table $H[\cdot]$ from a typical star network topology and its collected signal power table. $E[\cdot]$ in Fig. 7(b) is a snap shot of *SigPowerTable* in a simulated network when the transmission power of IEEE 802.15.4 is 0.28 Watt and 8 nodes are positioned in a circle whose radius approximates to 20 meters. Note that if $E[i, j]$ equals -1, it means that node i has not listened node j yet so that they are hidden from each other. For the ith row of $H[\cdot]$ in Fig. 7(c), the coordinator lists all j at the ith row of $H[\cdot]$ when $E[i, j]$ equals -1. In Fig. 7(c), for instance, N_5 is added to the 1st row of $H[\cdot]$ since $E[1, 5]$ = -1. By repeating this step for all active nodes, the coordinator finishes building $H[\cdot]$ as in Fig. 7(c).

After extracting the hidden node relation table $H[\cdot]$, *BuildHiddenGroup*() looks at $H[\cdot]$ row by row and builds the group table $G[\cdot]$ by separating detected hidden nodes into different groups. Fig. 8 shows how *BuildHiddenGroup*() carries out the grouping operation from a given $H[\cdot]$ in Fig. 8(a). Since N_1 and N_5 are hidden from each other, the coordinator first creates two groups G_1 and G_2 and put them in G_1 and G_2 separately as shown in Fig. 8(b). And then it writes down 1 in the 1st and 5th column of Check Table $C[]$ shown in Fig. 8(c) indicating that N_1 and N_5 are already assigned to their appropriate groups. Then it scans the second row of $H[\cdot]$ and finds out that N_2 is hidden from N_5 so that N_2 is joined at G_1 since G_1 does not contain N_5. By the same method, it allocates the remaining nodes into G_1, G_2, and G_3. Fig. 8(b) depicts the contents of $G[\cdot]$ after dispersing all hidden nodes except N_4. Note that three groups are enough to separate all hidden nodes in this example.

Finally, *CompleteGroup*() distributes the remaining nodes in a way to balance the group size for improving the performance. [17] reports that when the number of nodes is greater than 12, the throughput of IEEE 802.15.4 rapidly degrades. As portrayed in Fig. 8, *CompleteGroup*() preferably adds N_4 into G_3 since the size of G_3 is smaller than those of G_1 and G_2.

D. Allocation of Per-Group Time

To broadcast the group information, HAG adds two new fields named group identification (GI) for informing the assigned group identifier to each node and per-group time (GT) for notifying the allocated per-group time to each group as drawn in Fig. 9. GI field is further divided into one node count (NC) subfield and a variable number of pairs of node address (NA) and the associated GroupID subfields. NC indicates the number of pairs in the subsequent subfields. By reading GI field, each node recognizes which group it is assigned to.



N_1	N_5		
N_2	N_5		
N_3	N_6	N_7	N_8
N_4			
N_5	N_1	N_2	N_8
N_6	N_3		
N_7	N_3		
N_8	N_3	N_5	

(a)

G_1	N_1	N_2	N_3
G_2	N_5	N_6	N_7
G_3	N_8		

(b)

N_1	N_2	N_3	N_4	N_5	N_6	N_7	N_8
1	1	1	0	1	1	1	1

(c)

G_1	N_1	N_2	N_3
G_2	N_5	N_6	N_7
G_3	N_8	N_4	

(d)

Fig. 8: An example process for establishing hidden node relation table: (a) Hidden node relation table $H[\cdot]$, (b) intermediate group table, (c) check table $C[\cdot]$, and (d) complete group table $G[\cdot]$.

In a similar way, GT field is split into one group count (GC) subfield and a variable number of three subfileds, GroupID, group start time (GST) and group duration time (GDT). By scanning GT field, each node knows when they start to contend for the channel. Note that HAG also employs the deferment algorithm within each per-group time, meaning that when a node cannot finish its transmission within its per-group time, it defers the transmission to its next per-group time.

To maximize the performance, HAG algorithm adopts a dynamic time allocation scheme to readjust the per-group time according to each group's bandwidth demand. To dynamically adjust the per-group time, the coordinator periodically computes *GroupTime*$_i$ in (3) where *CAPtime*, *TrafficAmount*$_i$, and M represent the duration of CAP, the demand of group i for transmission time, and the number of established groups, respectively. For dynamic allocation of per-group times, the coordinator either increments or decrements *TrafficAmount*$_i$ based on whether *AvgColRate*$_i$ of group i exceeds the predetermined *ColThreshold*.

$$\mathit{GroupTime}_i = \frac{\mathit{TrafficAmount}_i}{\sum_{j=0}^{M} \mathit{TrafficAmount}_j} \times \text{CAP}_{\text{time}}. \qquad (3)$$

E. Implementation Issues

This subsection addresses two implementation issues of HAG algorithm such as time and space complexity and synchronization. At first, the time and space complexities of HAG algorithm are estimated to be $O(N^2)$ since it linearly scans a two-

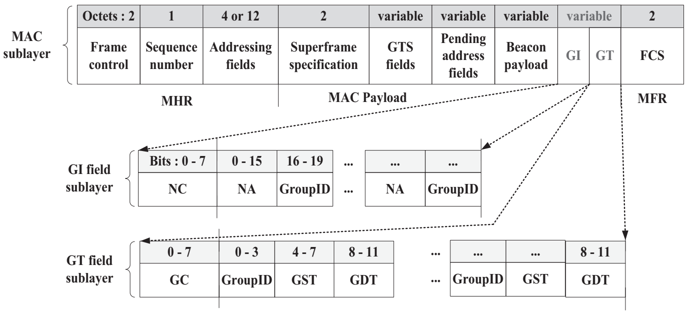


Fig. 9: Modified beacon message format for HAG algorithm.

dimensional $N \times N$ table in *ReGroup*() where N is the total number of nodes. Since it should look at each entry in this table to build hidden relations among nodes, its time and space complexities rise up to $O(N^2)$. As the number of nodes increases, we think that the current HAG algorithm would not be feasible. In near future, we plan to reduce these complexities down to $O(N\log N)$ by replacing the current linear search algorithm with a better one appropriate for our purpose.

Secondly, HAG algorithm requires nodes to be synchronized to send their frames only within their per-group duration. For synchronization, however, it does not need any additional synchronization mechanisms other than periodic beacon messages since we presume that beacon messages are broadcast frequently enough for nodes to align their clocks. We, furthermore, believe that HAG algorithm can operate efficiently even though clocks of nodes are loosely synchronized unless two adjacent per-group intervals are considerably overlapped. Note that the legacy IEEE 802.15.4 nodes run TDMA during CFP interval during which they should send their frames aligned with the predetermined time boundaries. We think that the amount of clock drift allowable in IEEE 802.15.4 would be enough accurate for nodes in a group not to interfere with transmissions from the neighbor groups.

V. ANALYTICAL MODEL

This section builds the analytical performance model for HAG algorithm based on the number of groups and the distribution of nodes in groups. At first, (4) represents the average throughput of HAG algorithm, $S_{\text{avg}}(N, M)$ when N nodes are dispersed into M groups under the assumption that each group is given with T_i per-group time. In (4), $S(G_1, G_2, \cdots, G_M)$ is the average throughput under a specific scenario when the sizes of M groups are $G_1, G_2, \cdots, G_M$ and their group times are $T_1, T_2, \cdots, T_M$ while $P(G_1, G_2, \cdots, G_M)$ is the probability for an arbitrary group distribution. The numerator for $P(G_1, G_2, \cdots, G_M)$ denotes the number of ways to make M groups out of N nodes whose sizes are $G_1, G_2, \cdots, G_M$ while the denominator represents the number of all possible ways to organize N nodes into M groups minus the number of cases where some groups have no node.

Table 3: Values of parameters for IEEE 802.15.4 simulations.

Parameter	Value
Packet payload	70 Bytes
MAC header	7 Bytes
PHY header	6 Bytes
ACK	11 Bytes
Channel bit rate	250 kbps
macMinBE	3
aMaxBE	5
Beacon order (BO)	3
Superframe order (SO)	3
Rx threshold	8.54×10^{-7} Watt
CS threshold	8.54×10^{-7} Watt

$$S_{\text{avg}}(N, M) = \sum_{G_1=1}^{N-(M-1)} \sum_{G_2=1}^{N-(M-2)-G_1} \cdots \sum_{G_{M-1}=1}^{N-(G_1+\cdots+G_{M-2})-1} S(G_1, \cdots, G_M) P(G_1, \cdots, G_M)$$

where

$$S(G_1 + \cdots + G_M) = \frac{1}{\sum_{i=0}^{M} T_j} (T_1 S(G_1) + \cdots + T_M S(N - (G_1 + \cdots + G_{M-1})))$$
$$P(G_1 + \cdots + G_M) = \frac{{}_N C_{G_1} \times {}_{N-G_1} C_{G_2} \times \cdots \times {}_{N-G_1-\cdots G_{j-1}} C_{G_M}}{M^N - \sum_{j=1}^{M-1} {}_M C_j \times {}_N C_{G_1} \times \cdots \times {}_{N-G_1 \cdots - G_{j-1}} C_{G_j}}. \quad (4)$$

Equation (5) finally shows the throughput of one group when it contains n nodes. Note that (5) is just the performance of an n-node IEEE 802.15.4 network with deferment algorithm as in Table 1

$$S(n) = \frac{n\tau(1-\alpha)(1-\beta) P_S L_{pl}}{(1-\tau) + \tau\alpha + 2\tau(1-\alpha) + \tau(1-\alpha)(1-\beta)(P_S T_S + (1-P_S) T_C)}. \quad (5)$$

VI. EXPERIMENTS

This section evaluates the performance of IEEE 802.15.4 with HAG algorithm when a coordinator relays data among nodes placed around it in the form of a ring while varying the number of hidden nodes. For simulations, we set the various parameters related to IEEE 802.15.4 as in Table 3 where RX threshold and CS threshold are the thresholds determining three ranges for noise, unrecognizable signal, and detectable signal, respectively. We also set the transmission signal power of each node to 8.54×10^{-7} Watt which enables signals to radiate within 15 meters in our simulation environments where two-ray ground model is assumed for signal attenuation [18].

To systematically manipulate the number of hidden nodes, our simulations adopt the star topology where we adjust three topological parameters, r, T, and θ in Fig. 10. The first two parameters indicate the maximum radiuses of radiation from the coordinator R_c and an arbitrary node R_s, respectively, while the

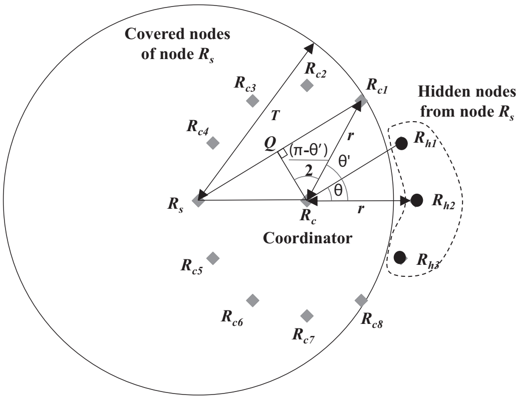


Fig. 10: Simulated network topology with hidden nodes.

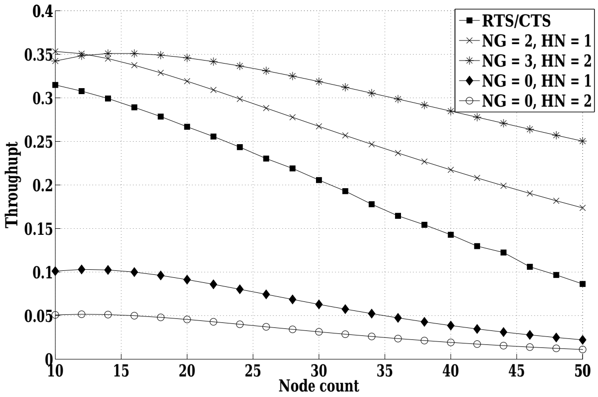


Fig. 11: Comparison of throughput between RTS/CTS mechanism and HAG algorithm.

last θ is the angle between two adjacent nodes. In Fig. 10, at the point of view of a sender, R_s, nodes ranging from R_{h1} and R_{h3} are hidden while other nodes from R_{c1} to R_{c8} are covered. To vary each group size and the number of hidden nodes, we adjust T and θ accordingly in below simulation experiments. For more details, please refer to [19].

Fig. 11 plots the theoretical throughput accomplished by RTS/CTS mechanism and HAG algorithm as a function of the number of nodes in a network where the number of HN are either 1 or 2, leading to different number of groups (NG) such as 2 or 3. Fig. 11 observes that RTS/CTS mechanism is inferior to HAG algorithm even though RTS/CTS mechanism gracefully alleviates the throughput fallout of the legacy IEEE 802.15.4. For Fig. 11, the sizes of RTS and data frames are set to 16 bytes and 70 bytes, respectively. HAG algorithm, for example, outperforms RTS/CTS mechanism at least by 23% when the number of nodes is 30. In detail, HAG algorithm outweighs RTS/CTS mechanism by 87% and 95% when the number of hidden nodes is 1 and 2 respectively in a network with 50 nodes.

Fig. 12 estimates the overhead of grouping nodes by plotting the throughput fluctuation of HAG algorithm measured in simulations while varying the number of nodes and the number of possible groups. It witnesses that fewer groups performs better in lightly populated networks while the more groups are better in dense the networks. As the network is congested, collisions become the dominant factor affecting overall throughput compared to the overhead incurred from the grouping operation. Note that the more groups, the more frames would be deferred due to the fragmentation of CAP duration.

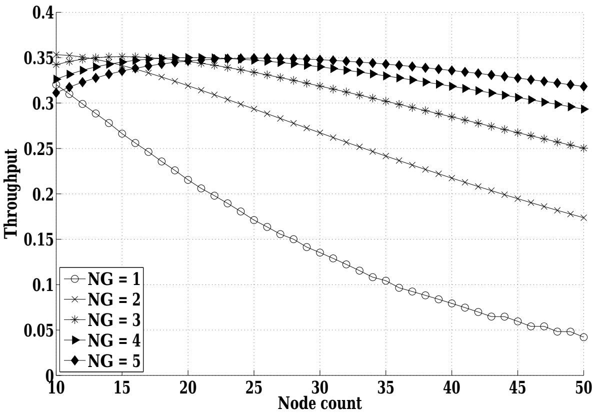


Fig. 12: Throughputs as a function of groups and nodes.

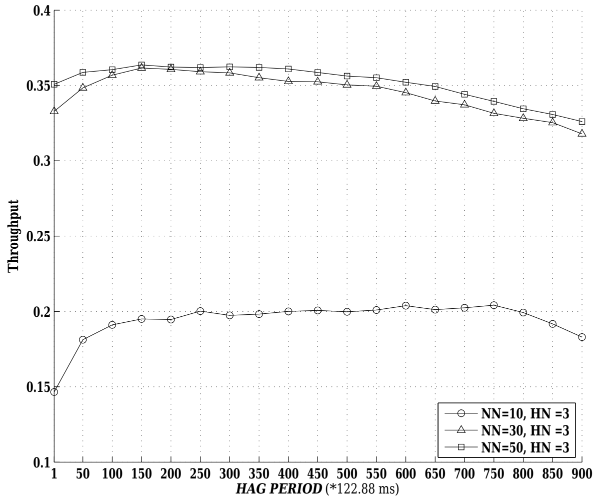


Fig. 13: Effects of *HAG_PERIOD* on performance of HAG algorithm.

Fig. 13 illustrates the effect of *HAG_PERIOD* on the performance of HAG algorithm in 10-node, 30-node, and 50-node networks with 3 hidden nodes. Note that *HAG_PERIOD* in Fig. 13 is counted as the multiples of superframes. Fig. 13 shows that the performance of HAG algorithm is not sensitive to the size of *HAG_PERIOD* in these static topologies unless *HAG_PERIOD* is set to be too small like one superframe or too large like 900 superframes. In our simulations, we set *HAG_PERIOD* to one superframe to show that HAG algorithm still can outperform IEEE 802.15.4 even under the harsh condition.

Figs. 14 and 15 display throughput fluctuations of HAG algorithm with either dynamic per-group time allocation (D-HAG) or static per-group time allocation (S-HAG) as a function of node count in a network with only one hidden node. In both

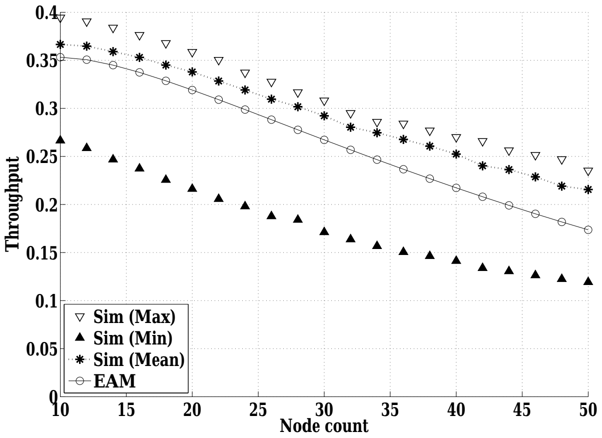


Fig. 14: Comparison of max, mean and min throughput of HAG algorithm with dynamic group time allocation.

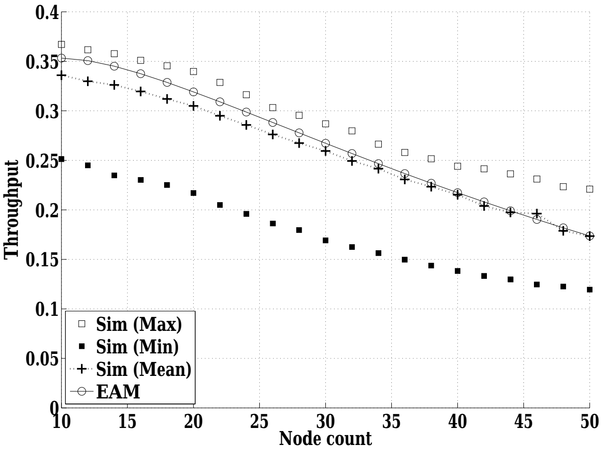


Fig. 15: Comparison of max, mean and min throughput of HAG algorithm with static per-group time allocation.

Figs. 14 and 15, on average the maximum throughput surpasses the minimum by around 40%. The comparison of the corresponding throughput in Figs. 14 and 15 shows that the maximum throughput of D-HAG outperforms that of S-HAG by around 7% while the minimum throughput of D-HAG is superior to that of S-HAG by 2%, approximately. The better performance of D-HAG is of course due to that D-HAG appropriately assigns larger transmission time to larger groups.

VII. CONCLUSIONS

This paper proposes a grouping algorithm to improve the performance of IEEE 802.15.4 networks, especially when the networks contain hidden nodes and are heavily congested. Based on the analytical model and simulations, it confirms that the grouping algorithm can achieve better performance by up-to 95% in severe conditions. It also presents an extended analytical model to accurately evaluate the behavior of IEEE 802.15.4 networks including the effect of the deferment technique under unsaturated environments.

As future research, we will investigate an appropriate search and sort algorithm for identifying hidden nodes out of a two-dimension array to lessen the time and space complexities of HAG algorithm down to $O(N \log N)$. We will also closely measure the trade-off between the grouping overhead and the mobility of nodes since HAG algorithm needs to reorganize groups when nodes move around. We, finally, will design a dynamic algorithm to adjust *HAG_PERIOD* according to the channel state's variability for better performance.

REFERENCES

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

[2] *Wireless medium access Control (MAC) and physical layer (PHY) specifications for low-rate wireless personal area networks(LR-WPANs)*, IEEE Standard 802.15.4-2006.

[3] P. K. Patro, M. Raina, V. Ganapathy, M. Shamaiah, and C. Thejaswi, "Analysis and improvement of contention access protocol in IEEE 802.15.4 star network," in *Proc. MASS*, 2007, pp. 1–8.

[4] Y. S. Shin, G. W. Hyun, J. S. Ahn, H. C. Kim, and K. W. Lee, "An analytical model for LR-WPAN performance in the presence of hidden nodes," *The KIPS Trans.: Part C*, no. 1, pp. 133–142, 2009.

[5] L. J. Hwang, S. T. Sheu, Y. T. Shih, and Y. C. Chen, "Grouping strategy for solving hidden node problem in IEEE 802.15.4 LR-WPAN," in *Proc. WICON*, July 2005, pp. 26–32.

[6] G. Bianchi, "Performance analysis of the IEEE 802.11 distributed coordination function," *IEEE J. Sel. Areas Commun.*, vol. 18, no. 3, Mar. 2000.

[7] I. Tinnirello, S. Choi, and Y. Kim, "Revisit of RTS/CTS exchange in high-speed IEEE 802.11 networks," in *Proc. WOWMOM*, 2005.

[8] J. Misic, V. B. Misic, and S. Shafi, "Performance IEEE 802.15.4 beacon enabled PAN with uplink transmissions in non-saturation mode-access delay for finite buffers," in *Proc. BROADNETS*, Oct. 2004, pp. 416–425.

[9] Y. S. Shin, K. W. Lee, and J. S. Ahn, "Exploring the feasibility of differentiating IEEE 802.15.4 networks to support health-care systems," *J. Commun. Netw.*, vol. 13, no. 2, pp. 132–141, Apr. 2011.

[10] S. Ray and D. Starobinski, "On false blocking in RTS/CTS-based multihop wireless networks," *IEEE Trans. Veh. Technol.*, vol. 56, no. 2, Mar. 2007.

[11] Z. Abichar, J. Chang, and D. Qiao, "Group-based medium access for next-generation wireless LANs," in *Proc. WOWMOM*, 2006, pp. 35–41.

[12] Z. Chen, C. Lin, H. Wen, and Hao Yin, "An analytical model for evaluating IEEE 802.15.4 CSMA/CA protocol in low-rate wireless application," in *Proc. AINAW*, May 2007, pp. 899–904.

[13] S. Pollin, M. Ergen, and S. C. Ergen, "Performance analysis of slotted carrier sense IEEE 802.15.4 medium access layer," in *Proc. GLOBECOM*, Nov. 2006.

[14] T. Park, T. Kim, J. Choi, S. Choi, and W. Kwon, "Throughput and energy consumption analysis of IEEE 802.15.4 slotted CSMA/CA," in *IEEE Electron. Lett.*, vol. 41, issue 18, pp. 1017–1019, Sept. 2005.

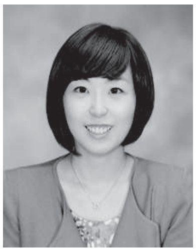
[15] E. C. Park and M. J. Rim, "Fair coexistence MAC protocol for contention-based heterogeneous networks," *The Comput. J.*, vol. 54 no. 8, pp. 1382–1397, Aug. 2011.

[16] *IEEE standard for local and metropolitan area networks-specific requirements, Part 11 : wireless LAN medium access control (MAC) and physical layer (PHY) specifications*, IEEE Standard 802.11-2007.

[17] R. K Patro, M. Raina, V. Ganapathy, M. Shamaiah, and C. Thejaswi, "Analysis and improvement of contention access protocol in IEEE 802.15.4 star network," in *Proc. MASS*, 2007, pp. 1–8.

[18] The NS-2 network simulator [Online]. Available: http://www.isi.edu/nsnam/ns/

[19] G. W. Hyun, Y. S. Shin, J. S. Ahn, K. W. Lee, "Analysis of effects of hidden nodes and CCA deferment algorithm on IEEE 802.15.4 performance using ns-2 simulator," *The KIPS Trans.: Part C*, vol. 16-C, no. 3, pp. 393–406, June 2009.



Jin-Yeong Um received the Ph.D., M.S., and B.S. degrees in Computer Engineering from Dongguk University in 2013, 2008, and 2004, respectively. Her major interests are wireless sensor networks, IEEE 802.11, IEEE 802.11e, IEEE 802.15.4, and network simulation techniques.

Jong-Suk Ahn received the Ph.D. and M.S. degrees in Electrical Engineering from the University of Southern California in 1995 and from the Korean Advanced Institute of Science and Technology in 1985, respectively. He also received the B.S. degree from Seoul National University in 1983. He was a Visiting Researcher in ISI/USC for one year in 2001. He was awarded the Gaheon Prize for the best paper in 2003 from the Korean Information Science Society. He is currently a Professor at Dongguk University in Korea. His major interests are sensor networks, dynamic error control and flow control algorithms over wireless Internet, network simulation techniques, IEEE 802.11, and IEEE 802.15.4.

Kang-Woo Lee received the M.S. and Ph.D. degrees in Electrical Engineering from the University of Southern California in 1991 and 1997, respectively. He also received the B.S. degree in Electronic Engineering from Yonsei University in 1985. He is currently a Professor at Dongguk University in Korea. His major interests are computer architecture, embedded systems, network computing, and simulation techniques.