Research Article

A Grouping Algorithm for Random Access Networks and Supermartingale Evaluation

Hongliang Sun
Xuefen Chi
Baozhu Yu

College of Information and Control Engineering, Jilin Institute of Chemical Technology, Jilin 132022, China
Department of Communications Engineering, Jilin University, Changchun 130012, China

Correspondence should be addressed to Hongliang Sun; shl915@163.com

Received 17 February 2022; Revised 7 August 2022; Accepted 20 August 2022; Published 9 September 2022

1. Introduction

In future communication networks, the wireless Internet of Things (IoT) communications are expected to support extremely massive terminals. The shortage of spectrum resource has always been an important issue. For large numbers of terminals with random arrival traffic, it is not suitable to reserve resource for individual terminals. Obviously, random access strategy can share the bandwidth resource better. The Carrier Sense Multiple Access/Collision Avoidance (CSMA/CA) mechanism is widely used in the random access protocol of IoT communications [1, 2]. However, it is well known that CSMA/CA protocol suffers from severe hidden node collision with the increasing number of terminals. The collision could result in packet loss, energy consumption, and performance degradation. Therefore, it is an inevitable problem to mitigate hidden terminals for CSMA/CA networks.

Much research has been devoted to settling the hidden node issue in random networks. The main methods include busy tone mechanism, Request-To-Send/Clear-To-Send mechanism, carrier-sense tuning mechanism, interference cancellation mechanism, and node grouping mechanism [3]. In contrast, the grouping-based approach is more applicable to networks with vast quantities of terminals. For the grouping mechanism, the nodes that have hidden node relationship are distributed into different groups as much as possible. In [4], the authors adopt the time difference method to determine the matrix of hidden node relationship, and the groups of hidden node avoidance are formed according to the matrix. The authors of [5] propose a Signal Strength Assistant (SSA) grouping scheme to dispose hidden node problem. The implementation of the algorithm is aided by the reference nodes with known location. In [6], the authors propose a hidden node grouping algorithm based on IEEE 802.11ah Restricted Access Window, in which the hidden node information is collected by the association process. Utilizing the integer programming optimization method, the authors of [7] propose a grouping scheme of hidden terminal avoidance. In [8], the intercluster grouping method is proposed to eliminate the hidden node problem. In [9], the authors propose a hidden terminal aware clustering mechanism, and the clusters are formed based on the detected information of the neighboring phase. The
grouping access strategy can significantly mitigate the effect of hidden nodes. However, less attention is paid to load balance when using grouping approaches to settle the hidden node issue.

Groups with balanced traffic could improve the efficiency guarantee for Quality of Service (QoS), because the service resource does not need to be drastically adjusted to match the requirement of each group. Moreover, the performance of balanced groups would not differ greatly with the same service, which facilitates the fairness guarantee in groups. Therefore, load balance should be taken into account when grouping. In [10], the authors realize load balanced grouping by constructing an optimization problem, in which the different traffic requirement of the nodes is considered. Based on the heuristic approach, the authors of [11] propose a coverage aware load-balanced clustering algorithm. The authors of [12] propose a cluster-based mobility load balance algorithm for heterogeneous cellular networks. In [13], the clustering method with an evolutionary approach is introduced aiming at the load balance. For the above research, only the load balance is considered, and the scheme is not applicable to CSMA/CA networks.

In most of the existing grouping algorithms, processing hidden terminals and achieving load balance are considered separately. Relying on the grouping strategy, the authors of [3] propose an H-NAMe mechanism to avoid hidden node collision. In the algorithm, only when a node could select different groups under the grouping principle, the node is considered to be divided into group with fewer nodes to promote load balance. The authors of [14] propose a hidden node aware grouping (HAG) algorithm, in which per-group time is allocated according to the bandwidth demand for load balance. The load balance is settled after the grouping process is executed. It is highly anticipated that an algorithm could perform hidden node avoidance and load balance simultaneously.

The game theory is a powerful mathematical tool used to make grouping decisions. The authors of [15] propose a game theoretic clustering algorithm for the nonuniform distributed nodes. The algorithm could effectively balance the number of nodes in each cluster, thus prolonging the effective lifetime of networks. Based on the distributed nature of the M2M networks, the noncooperative power management game is formulated in [16], and the optimal charging transmission powers of the cluster heads are derived. To implement energy-efficient cluster head selection and routing, the authors of [17] propose a game-based clustering and multihop routing scheme. For grouping with multiple constraints, it also could be achieved through the optimization problem. However, the solution would be difficult to be solved with the increasing number of nodes. In this paper, according to the characteristics of CSMA/CA networks, we construct the payoff function that the gain of hidden node reduction and load balance is considered together. By introducing the game theory, the grouping can be realized by adaptive adjustment.

Benefiting from the grouping strategy, the hidden terminal issue is mitigated, and the QoS performance would be enhanced. When the grouping is completed, the reasonable performance evaluation is conducive to the resource management. Delay is one of the most significant QoS indicators, but the delay analysis has always been a notorious problem for CSMA/CA mechanism. The large deviation is a typical method for delay QoS evaluation, while the methodology proved to be conservative. Recently, the supermartingale theory demonstrates an advantage in delay analysis, and the more accurate results of delay performance are derived under CSMA/CA service in [18]. Capitalizing on the supermartingale model, the end-to-end delay performance is evaluated for multihop vehicular ad hoc networks and multimedia heterogeneous high-speed train networks in [19, 20], and the delay QoS bounds are proved remarkably tight by simulations. In our previous work, we have estimated the bandwidth requirement for aggregate traffic under delay QoS constraint based on supermartingale theory, which measures bandwidth more accurately [21]. In this paper, we evaluate the delay performance for the completed groups relying on the supermartingale analysis framework. To better estimate the overall delay performance after grouping, we set up a new QoS indicator that is the average delay-violation probability bound.

The main contributions of this paper are summarized as follows:

1. We propose a grouping algorithm based on the game theory for random access networks. The payoff function of each node is defined by taking into account the overall income. Through the game adjustment, the hidden node collision is mitigated and the load balance is achieved simultaneously. The proposed grouping algorithm could improve QoS performance and resource utilization.

2. We introduce the supermartingale theory to analyze the delay performance. The supermartingale parameters are derived, and a new QoS indicator is established to evaluate the grouping algorithm. The delay analysis results could provide a reference for service resource management.

The remainder of this paper is organized as follows. Section 2 formulates the grouping problem and the grouping algorithm is designed. In Section 3, we utilize the supermartingale theory to analyze delay performance for the completed groups. Simulation results demonstrate the advantage of this research in Section 4. Finally, we conclude this paper in Section 5.

2. Grouping Algorithm

In this section, our grouping problem is mathematically formulated, and then, the detailed grouping algorithm is described. The main notations and descriptions used in this section are listed in Table 1.

2.1. Problem Formulation. The simplified scenario under study is as Figure 1. The location of nodes is fixed, and the hidden node relationship does not change. The arrival traffic of each node may be different. The access point (AP) is the
Table 1: The main notations and descriptions of grouping algorithm.

| Notations | Descriptions |
|-----------|--------------|
| $N$       | The number of terminals |
| $\alpha$  | The level of hidden nodes |
| $a_i(n)$  | The number of arrival packets from node $i$ in slot $n$ |
| $\lambda_i$ | The arrival intensity of node $i$ |
| $K$       | The number of groups |
| $L_k$     | The total traffic load of group $k$ |
| $\Gamma$  | The set of participants |
| $X_i$     | The strategy space for node $i$ |
| $U_{h,i}$ | The gain of hidden node minimization |
| $U_{j,i}$ | The gain of load balance |
| $H_{c,now}$ | The number of hidden nodes in current group for node $i$ |
| $H_{c,exp}$ | The number of hidden nodes in target group for node $i$ |
| $L_{c,max}$ | The maximum total traffic load after adjustment |
| $L_{c,min}$ | The minimum total traffic load after adjustment |
| $U_{\text{all}}$ | The overall income when node $i$ is adjusted |
| $U_{\text{last}}$ | The final income of each node |

The primary purpose of grouping is to reduce hidden terminals, so the nodes that have hidden relation should be divided into different groups as much as possible. In addition, the load balance of traffic should be maintained to the greatest extent while mitigating hidden terminals. In this model, the load balance of traffic means that the sum of arrival intensity in each group is approximate. The scenario we study is mainly aimed at terminals with relatively stable position and traffic characteristics. After the mechanism runs stably, the groups do not need to be adjusted frequently.

2.2. Algorithm Design. In this part, we design a grouping algorithm based on game theory. Initially, the groups are formed only by considering load balance. Then, the algorithm performs multiple rounds of game adjustment. The principle of adjustment is to reduce hidden node pairs while maintaining load balance. We take $K$ groups as an instance to illustrate the grouping process.

The initial groups are formed according to the traffic load of each node. Firstly, $K$ nodes are selected arbitrarily and distributed to the $K$ groups. Then, the remaining nodes are divided into groups one by one according to the following regulation: The selected node is placed in the group with the smallest total traffic load. The total traffic load of group $k$ is as follows:

$$L_k = \sum_{i=1}^{m} \lambda_{k,i},$$

where $m$ represents the number of nodes in group $k$ and $\lambda_{k,i}$ represents the arrival intensity of node $i$ in group $k$. The initial grouping ensures that the difference of traffic load in each group is not too large.
The initial grouping is performed by AP. After initial grouping, AP broadcasts the hidden node relationship, the node information, and the initial grouping results. Subsequently, the game adjustment algorithm is executed.

In this grouping mechanism, a round of game $G$ is the process that the nodes attempt to regroup. The set of participants $\Gamma$ in game $G$ consists of all nodes, and $\Gamma = \{1, 2, \cdots, N\}$. Each node could be viewed as a player, and they could choose to remain static or join other groups while gaming. The strategy space for node $i$ is as follows:

$$X_i = [x_{i1}, x_{i2}, \cdots, x_{ik}], i \in \Gamma,$$

when node $i$ joins in group $k$ the decision variable $x_{ik}$ is 1; otherwise it is 0. The gain consists of two parts: the gain of hidden node minimization $U_{h,i}$ and the gain of load balance $U_{l,i}$.

$$U_{h,i} = H_{i,\text{now}} - H_{i,\text{exp}}$$

$$U_{l,i} = e^{-(L_{i,\text{max}} - L_{i,\text{min}})}.$$

$H_{i,\text{now}}$ indicates the number of hidden terminals in current group for node $i$. $H_{i,\text{exp}}$ indicates the number of hidden terminals in target group for node $i$, and $H_{i,\text{exp}} = \min [H_{i,k}]$. $H_{i,k}$ is the number of hidden terminals for node $i$ after joining in group $k$. The group that has the minimum $H_{i,k}$ is the target group for node $i$. If the node is adjusted to a nontarget group, the $U_{h,i}$ is 0. $L_{i,\text{max}}$ is the maximum total traffic load after adjustment, and $L_{i,\text{max}} = \max [L_{i,k}]$. $L_{i,\text{min}}$ is the minimum total traffic load after adjustment, and $L_{i,\text{min}} = \min [L_{i,k}]$.

For the two gains, the hidden node minimization gain plays a decisive role in whether the node is adjusted. We define the overall income when node $i$ is adjusted, which is as follows:

$$U_{i,\text{all}} = I(U_{h,i})U_{l,i},$$

where $I(\bullet)$ is an indicator function, and $I(U_{h,i}) = \begin{cases} 1 & U_{h,i} > 0, \\ 0 & U_{h,i} \leq 0. \end{cases}$

Through (8), we know that the overall income exists only when $U_{h,i}$ exists. The value of $U_{i,\text{all}}$ depends on $U_{l,i}$.

We define the payoff function for each node as follows:

$$U_{i,\text{last}} = \begin{cases} \frac{U_{i,\text{all}}}{N} & \text{The node with the maximum } U_{i,\text{all}} \text{ is adjusted,} \\ 0 & \text{otherwise}. \end{cases}$$

For this income distribution, maximizing $U_{i,\text{all}}$ means that each participant can obtain the maximum income. This cooperation mode could maximize the overall income and make the cooperation stable.

In the following, we will introduce the detailed process for the game adjustment. Firstly, $H_{i,\text{now}}$ and $H_{i,k}$ of each node are calculated. Correspondingly, the target group and $H_{i,\text{exp}}$ are determined. Next, the nodes that meet the following condition are counted:

$$H_{i,\text{now}} - H_{i,\text{exp}} > 0.$$ (10)

If none of the nodes satisfies the condition, the game ends. That is, all nodes have the minimum number of hidden node pairs in current group, and the adjustment would not result in the reduction of hidden terminals. Otherwise, the nodes that satisfy inequality (10) calculate the load balance gain $U_{l,i}$ and calculate the overall gain $U_{i,\text{all}}$ further. We assume that there are $m *$ nodes satisfying this condition. According to the definition of $U_{i,\text{last}}$, except for the $m *$ nodes to be adjusted, the gain of other nodes would be 0. Then, the $m *$ nodes send the gain information to AP. AP determines the node that could bring the largest $U_{i,\text{last}}$ and adjusts it to target group. If there are multiple nodes with the same largest $U_{i,\text{last}}$, randomly select one node and adjust it in target group. Finally, $H_{i,\text{now}}$ and $H_{i,k}$ of each nodes are recalculated, and a new round of game starts.

During game $G$, every adjustment could reduce the number of hidden terminals. The same game needs to be played many rounds. The grouping results could be represented as follows:

$$X_i = [x_{i1}, x_{i2}, \cdots, x_{ik}], i \in \Gamma.$$

When node $i$ belongs to group $k$, the decision variable $x_{ik}$ is 1; otherwise, it is 0.

Obviously, more groups are divided, the less the number of hidden terminals in each group. The number of groups could be determined relying on the level of the hidden nodes. We could define the threshold $\eta$ as the acceptable average level of hidden nodes. When grouping is completed, the average ratio of hidden node pairs $\bar{a}$ should be satisfied $\bar{a} < \eta$. If $\bar{a}$ does not meet the expectation, we can add the number of groups. However, too many groups will impose a burden on system management. Therefore, the optimal number of groups would be determined according to different situations.

The grouping program can be described as the following algorithm 1:

The grouping process is based on the game theory. It is necessary to illustrate the existence of Nash equilibrium and the astringency of the game algorithm.

**Theorem 1.** There exists a Nash equilibrium solution for game $G$, and the game-based grouping algorithm is convergent.

The proof can be seen in Appendix A.

2.3. Algorithm Complexity Analysis. The algorithm is coordinated by central AP and distributed nodes. AP is responsible for integrating information and making decisions, and the main computation task is transferred to each node.
Obviously, the computation task is heavier with more nodes. Nonetheless, each node has computation capacity, and the computational complexity will not increase extremely with the increasing number of nodes. The number of groups is the key factor affecting the computational complexity.

For initial grouping, the computational complexity of each node is \(O(N)\), where \(O(\cdot)\) indicates complexity. Then, we analyze the computational complexity for each node in a round of game. In the best case, if only one node adjustment reduces the hidden terminals, the complexity is \(O(N)\). In the worst case, if the adjustment of each node reduces the hidden terminals, the complexity is \(O(N^2)\).

### 3. Delay Performance Evaluation

In this section, we will analyze the performance of statistical delay QoS for the completed groups. The analysis results could not only evaluate the grouping effect but also provide guidance for service resource management. The main notations and descriptions used in this section are listed in Table 2.

We first conduct delay analysis for the packet transmission process of a single node. The queuing system with Poisson arrival and CSMA/CA service is analyzed. The supermartingale method is introduced to evaluate the delay performance. We transform the processes of arrival and service to supermartingale processes and derive the bound of delay-violation probability relying on stopping time theory.

Before performing the supermartingale analysis, we first introduce the definitions of supermartingale, arrival-martingales, and service-martingales, which are as below.

**Definition 2** (See [22]) (supermartingale). \(\forall n \geq 0\), the random process \(\{Z(n), n \geq 0\}\) is a supermartingale if

\[
E[|Z(n)|] < \infty
\]

**Table 2:** The main notations and descriptions of delay analysis.

| Notations | Descriptions |
|-----------|--------------|
| \(p\)     | The transition probability from backoff state to transmission state |
| \(q\)     | The transition probability from transmission state to backoff state |
| \(R\)     | The transmission rate in transmission state |
| \(a(n)\)  | The instantaneous arrival |
| \(A(n)\)  | The cumulative arrival |
| \(s(n)\)  | The instantaneous service |
| \(S(n)\)  | The cumulative service |
| \(Q(n)\)  | The queue length |
| \(d(n)\)  | The delay |
| \(D\)     | The target delay |
| \(B_i\)   | The delay-violation probability bound for node \(i\) |
| \(B_k\)   | The delay-violation probability bound for node \(i\) of group \(k\) |
| \(\overline{B}_k\) | The average delay-violation probability bound of group \(k\) |
| \(h_a(\cdot), h_s(\cdot), K_a, K_s, \theta\) | The supermartingale parameters |

\[E[Z(n + 1)|Z(0), Z(1), \ldots, Z(n)] \leq Z(n)\]

**Definition 3** (See [18]) (arrival-martingales). The arrival process \(A\) admits arrival-martingales if for every \(\theta > 0\), there is a \(K_\theta \geq 0\) and a function \(h_a : rng(a) \rightarrow R^+\) such that the process

\[\text{Input: The number of groups } K, \text{ the arrival intensity } \lambda, \text{ the hidden node matrix } W.\]

### Algorithm 1: The game-based grouping algorithm.

1. Select arbitrary \(K\) nodes and assign them to the \(K\) groups respectively.
2. Calculate the traffic load of each group, and determine the group \(M\) that has the minimum traffic load.
3. Assign the next node \(i\) to group \(M\).
4. If node \(i\) is the last node
5. Go to step 9.
6. Else
7. Go to step 2.
8. End If
9. Calculate \(H_{\text{now}}\) and \(H_{\text{exp}}\) for each node.
10. Determine the \(m^*\) nodes that satisfy inequality (10).
11. If \(m^* = 0\).
12. Stop.
13. Else
14. Calculate \(U_{\text{all}}\) for the \(m^*\) nodes.
15. Adjust the node that could bring the largest \(U_{\text{last}}\) to target group.
16. Go to step 9.
17. End If
\[ h_s(a(n))e^{\theta(A(n)-nK_s)} \leq 0 \]

is a supermartingale.

**Definition 4** (See [18]) (service-martingales). The service process \( S \) admits service-martingales if for every \( \theta > 0 \), there is a \( K_s \geq 0 \) and a function \( h_s : rng(s) \rightarrow R^+ \) such that the process

\[ h_s(s(n))e^{\theta(nK_s-S(n))}, n \geq 0 \]

is a supermartingale.

In the definitions, \( rng(\cdot) \) indicates the domain operator. \( a(n) \) represents the instantaneous arrival, and \( A(n) \) represents the cumulative arrival. \( s(n) \) represents the instantaneous service, and \( S(n) \) represents the cumulative service. \( \theta \) is an attenuation index, and the parameters \( K_s, h_s(\cdot), K_s \), and \( h_s(\cdot) \) all depend on \( \theta \).

The arrival process of each node obeys the Poisson process. The supermartingale relative to Poisson arrival could be constructed through exponential transform [21]. Then, we give the supermartingale parameters of (12) for Poisson arrival. Poisson arrival is an independent and identically distributed increment process, and we can take \( h_s(a(n))=1 \). \( K_s \) could be determined according to the following formula:

\[ E[\theta(A(n))] = e^{\theta K_s}. \]

The same CSMA/CA mechanism is adopted for each group. We use the CSMA/CA model from [18]. The Markov chain depicting the service process is shown in Figure 2. The state \( i \) refers to the node \( i \) in the transmission state, and all sources are in backoff mode in state 0. \( p \) is the transition probability from backoff state to transmission state, and \( q \) is the transition probability from transmission state to back-off state. \( R \) is the transmission rate in transmission state.

The transition matrix for the service process is given as

\[
V = \begin{pmatrix}
1 - p & p & \cdots & p \\
q & 1 - q & \cdots & 0 \\
& \vdots & \ddots & \vdots \\
q & 0 & \cdots & 1 - q
\end{pmatrix}.
\]

Then, we give the supermartingale parameters of (13) for this service process. Define the exponential column-transform matrix \( (V^\theta) \) as

\[
V^\theta = \begin{pmatrix}
1 - p & p & \cdots & p e^{\theta R} \\
q & 1 - q & \cdots & 0 \\
& \vdots & \ddots & \vdots \\
q & 0 & \cdots & (1 - q)e^{\theta R}
\end{pmatrix}.
\]

The queue length could be shown as below.

\[ Q(n) = \sup_A \{A(n) - S(n)\}, \]

where \( \Lambda \) denotes equality in distribution. According to the Little’s Law, the delay \( d(n) \) could be denoted as

\[ d(n) = \frac{Q(n)}{\lambda_f}. \]

For the supermartingale process, the expectation at a stopping time is less than or equal to the expectation of its initial value. Through the stopping time theory, we can obtain the bound of delay-violation probability [21], which is shown as Theorem 5.
Theorem 5. For any \( D > 0 \), the following delay-violation probability bound holds:

\[
P\{d(n) \geq D\} \leq B_f = \frac{E[h_s(s)]E[h_s(s)]}{H} e^{-\lambda_d D},
\]

where

\[
H = \min \left\{ h_a(a(n))h_s(s(n)) : a(n) - s(n) > 0 \right\},
\]

\[
\theta' = \sup \{ \theta > 0 : K_s = K_i \}.
\]

\( B_f \) represents the delay-violation probability bound for node \( J \).

The proof can be seen in Appendix B.

To satisfy the requirement of statistical delay QoS, the delay-violation probability \( P\{d(n) \geq D\} \) should be less than a threshold when the target delay value \( D \) is given. The results derived in (20) could assist to judge whether the delay requirement is satisfied. Meanwhile, the formula could reflect the delay performance.

We establish a new QoS indicator to analyze the delay performance for the completed groups, which is the average delay-violation probability bound of group \( k \):

\[
\overline{B}_k = \frac{\sum_{i=1}^{m} B_{k,i}}{m},
\]

where \( B_{k,i} \) is the delay-violation probability bound for node \( i \) of group \( k \). This indicator can reflect the average level of delay performance for each group.

4. Results and Discussion

In this section, we will present the simulation results for the grouping algorithm, and the performance of the scheme is evaluated. 50 nodes that the arrival traffic obeys Poisson distribution are simulated, and the hidden node matrix is simultaneously generated. The arrival intensity of each node is randomly generated from 6 to 9 packets/slot. The ratio of hidden node pairs is 20%. The number of groups \( K \) is set to 1–10. The target delay \( D \) is taken as 10–100 slots. All results are implemented by MATLAB.

We first investigate the effectiveness and advantages of the proposed algorithm. Figure 3 demonstrates the relationship between the number of groups and the average level of hidden nodes. For the proposed algorithm, the ratio of hidden node pairs decreases rapidly with the increasing groups, which reveals that the grouping strategy has a great effect on solving hidden node problem.

The hidden matrix-based regrouping (HMR) algorithm is proposed in [4], and it has a similar scenario to our research. HMR algorithm first finds hidden node pairs and generates a hidden node matrix and then regroups nodes to reduce collision using the hidden node matrix. In our work, we focus on the grouping process after the hidden node relationship is detected, and load balance is considered when grouping. We perform a comparison with the hidden node regrouping algorithm, and it is called the existing algorithm in the following. As is presented in Figure 3, the average level of hidden nodes of the proposed algorithm descends faster than that of the existing algorithm, which illustrates that the game-based algorithm is more effective in mitigating hidden terminals. When the group number is more than 4, the ratio of hidden node pairs is rare. In the subsequent analysis, we take 4 groups as an example.

In the grouping algorithms of [4–9], only the hidden node problem is considered. In our work, the load balance is synchronously considered. Figure 4 displays the results after game adjustment. Not only the number of hidden node pairs is small, but also the load balance is realized. Figure 5 manifests the grouping results of the existing algorithm. Compared with the proposed algorithm, the number of hidden node pairs is more, and the difference in traffic load is larger. The results demonstrate that the proposed algorithm could settle the hidden node problem more effectively, and achieve load balance simultaneously.

In the grouping algorithms of [10–13], only the load balance is considered. In our algorithm, the load balance is not the main income in the grouping process, so the load balance effect may not be superior to other research. In relative terms, the advantage of this algorithm is to obtain two kinds of income simultaneously. The initial grouping algorithm is formed according to the traffic of each node, and it is an algorithm that only considers load balance. Figure 6 presents the results of the initial grouping algorithm. Although the load balance is implemented, there exist many hidden node pairs.
Then, we analyze the delay performance of each group. The average delay-violation probability bound is used as an indicator to evaluate delay performance. In Figure 7, the results present the delay performance curves for the groups with the best delay performance and the worst delay performance. It is shown that the average delay-violation probability bound decreases with the increasing target delay. The greater target delay corresponds to the looser delay requirement, so the QoS violation is not easy to occur and the average delay-violation probability bound is smaller. There is a small difference in delay performance between the two groups. To further guarantee the trade-off of the groups, we could allocate a larger available resource to the group with the large average delay-violation probability bound. Also, we compare the delay performance between the...
The results manifest that the ratio of hidden nodes increases rapidly with the increasing ratio of hidden nodes. Figure 9, we set the acceptable average level of hidden nodes is the number of partitioned groups. In the simulation of grouping strategy, and other methods of hidden node avoidance is too high, we could not merely rely on the algorithm, we know that the main factor affecting the complexity becomes larger with the increasing range of arrival intensity. For a large difference in arrival intensity, if we perform grouping directly, the nodes with heavy traffic are prone to congestion. We could assign the terminals with small difference in traffic load into one cluster and then group the nodes for the cluster.

From the computational complexity analysis of the algorithm, we know that the main factor affecting the complexity is the number of partitioned groups. In the simulation of Figure 9, we set the acceptable average level of hidden nodes as 10%. As expected, the required number of groups increases rapidly with the increasing ratio of hidden nodes. The results manifest that the ratio of hidden nodes has a significant effect on the algorithm complexity. If the level of hidden nodes is too high, we could not merely rely on the grouping strategy, and other methods of hidden node avoidance should be considered synchronously.

5. Conclusion and Future Work

In this paper, we proposed a grouping algorithm for random access networks based on the game theory. The income of hidden node minimization and load balance was considered simultaneously. Moreover, the supermartingale theory was introduced to analyze the delay performance, and a new QoS indicator was established to support the delay analysis. The proposed algorithm could mitigate hidden terminal collision, achieve traffic load balance, and facilitate the QoS guarantee efficiently. The analysis results could provide a reference for the allocation of service resource. However, when the traffic difference is large and the number of terminals is small, the grouping algorithm will not get a good result for load balance. In addition, if the traffic and location of terminals are not fixed, the updating mechanism needs to be explored. To ensure the QoS requirement of individual terminals, the scheduling scheme after grouping also needs to be studied. We are interested in investigating the above issue in future work.

Appendix

A. The Proof of Theorem 1

Proof. According to the Nash equilibrium existence theorem, in a game \( G \) with \( N \) participants, if \( N \) is limited and strategy space of the participants is a finite set, then there is one Nash equilibrium solution at least, but it may include mixed strategy [15]. Since the game \( G \) in this algorithm is a finite pure strategy game, there must be a Nash equilibrium solution. □

Assume that the state space composed of the Nash equilibrium state is \( \Omega^* \), and it is easy to know that \( \Omega^* \) is a finite set. A Nash equilibrium solution is based on the previous Nash equilibrium solution, and the conditional probability distribution of the upcoming state only depends on current state, so the algorithm constitutes a homogeneous Markov chain with finite state. Let \( \Omega^{**} \) denote all optimal Nash equilibrium sets. From the evolution of the algorithm, we have \( \Omega^{**} \subset \Omega^* \), and \( \Omega^{**} \) is a closed set. The state in \( \Omega^{**} \) is a recurrent state, and the state in \( \Omega' = \Omega^* - \Omega^{**} \) is a nonrecurrent state. The Markov chain will transform from any nonrecurrent state to \( \Omega^{**} \) in probability 1. Therefore, the algorithm is convergent.

B. The Proof of Theorem 5

Proof. According to the supermartingales relative to the arrival process and the service process, we can construct a new supermartingale relative to queue length, which is as the following:

\[
M_Q(n) = h_a(a(n))h_s(s(n))e^{\theta(A(n) - nK_s + nK_r - S(n))}. \tag{B.1}
\]

Let \( \theta \) be as defined. \( M_Q(n) \) could be converted as follows:

\[
M_Q(n) = h_a(a(n))h_s(s(n))e^{\theta(A(n) - S(n))}. \tag{B.2}
\]

Define the stopping time \( T \) as the first time when \( A(n) - S(n) \) exceeds \( \sigma \).

\[
T = \min \{ n : A(n) - S(n) \geq \sigma \}. \tag{B.3}
\]

Note that \( T = \infty \) is possible and

\[
P(Q(n) \geq \sigma) = P(T < \infty). \tag{B.4}
\]
Define the threshold $H$ as follows:

$$H = \min \{ h_a(a(n))h_y(s(n)) : a(n) - s(n) > 0 \}. \quad (B.5)$$

According to the stopping time theory of supermartingale, we have

$$E[h_a(a(0))|E[h_y(s(0))]] = E[M_Q(0)] \geq E[M_Q(T \land n)1_{\{T \leq n\}}]$$

$$(T \land n = \min \{ T, n \} (n \geq 0)) = E\left[h_a(a(T))h_y(s(T))e^{\theta(A(T)-S(T))}1_{\{T \leq n\}}\right]$$

$$\geq He^{\theta \alpha}P(T \leq n). \quad (B.6)$$

When $n \to \infty$, we have

$$P(Q(n) \geq \sigma) \leq \frac{E[h_a(a(0))|E[h_y(s(0))]]}{H} e^{-\theta \alpha}. \quad (B.7)$$

According to the Little’s Law, the bound of delay-violation probability is obtained as follows:

$$P(d(n) \geq D) \leq \frac{E[h_a(a(0))|E[h_y(s(0))]]}{H} e^{-\theta \lambda D}. \quad (B.8)$$

Thus, Theorem 5 holds.

**Data Availability**

The data could be available from the corresponding author on reasonable request.

**Conflicts of Interest**

The authors declare that they have no conflicts of interest.

**References**

[1] L. J. Terng and F. B. Hashim, “QoS provisioning in CSMA/CA-based opportunistic random access for WLAN,” in 12th IEEE Malaysia International Conference on Communications (MICC), pp. 333–338, Kuching, 2015.

[2] A. Mazin, M. Elkourdi, and R. D. Gitlin, “Comparison of slotted aloha-NOMA and CSMA/CA for M2M communications in IoT networks,” in 88th IEEE Vehicular Technology Conference (VTC-Fall), Chicago, 2018.

[3] A. Koubba, R. Severino, M. Alves, and E. Tovar, “Improving quality-of-service in wireless sensor networks by mitigating ‘hidden-node collisions’,” IEEE Transactions on Industrial Informatics, vol. 5, no. 3, pp. 299–313, 2009.

[4] S. G. Yoon, J. O. Seo, and S. Bahk, “Regrouping algorithm to alleviate the hidden node problem in 802.11ah networks,” Computer Networks, vol. 105, pp. 22–32, 2016.

[5] L. Zhang, H. Li, Z. Guo, L. Ding, F. Yang, and L. Qian, “Signal strength assistant grouping for lower hidden node collision probability in 802.11ah,” in 9th International Conference on Wireless Communication and Signal Processing (WCSP), Nanjing, 2017.

[6] R. Y. Wang and M. Lin, “Restricted access window based hidden node problem mitigating algorithm in IEEE 802.11ah networks,” IEICE Transactions Communications, vol. E101B, no. 10, pp. 2162–2171, 2018.

[7] H. Mosavat-Jahromi, Y. Li, and L. Cai, “A throughput fairness-based grouping strategy for dense IEEE 802.11ah networks,” in 2019 IEEE 30th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC), pp. 1–6, Istanbul, Turkey, 2019.

[8] R. Ravi, R. Samikannu, S. Venkatachary, and B. Diarra, “Intercluster link management for quality of service development by mitigating the hidden node problem in wireless sensor networks,” Journal of Testing and Evaluation, vol. 47, no. 6, pp. 3851–3863, 2019.

[9] S. H. Rhee and X. Lei, “Hidden terminal aware clustering for large-scale D2D networks,” Wireless Pers Commun, vol. 107, no. 3, pp. 1367–1381, 2019.

[10] T. C. Chang, C. H. Lin, K. C. J. Lin, and W. T. Chen, “Load-balanced sensor grouping for IEEE 802.11ah networks,” in IEEE Global Telecommunications Conference (GLOBECOM), San Diego, 2015.

[11] S. Singh and R. M. Sharma, “Heuristic based coverage aware load balanced clustering in WSNs and enablement of IoT,” International Journal of Information Technology and Web Engineering, vol. 13, no. 2, pp. 1–10, 2018.

[12] M. M. Hasan and S. Kwon, “Cluster-based load balancing algorithm for ultra-dense heterogeneous networks,” IEEE Access, vol. 8, pp. 2153–2162, 2020.

[13] F. Dehestani and M. A. Jabreil Jamali, “Load balanced clustering based on imperialist competitive algorithm in wireless sensor networks,” Wireless Personal Communications, vol. 112, no. 1, pp. 371–385, 2020.

[14] J. Y. Um, J. S. Ahn, and K. W. Lee, “Evaluation of the effects of a grouping algorithm on IEEE 802.15.4 networks with hidden nodes,” Journal of Communications and Networks, vol. 16, no. 1, pp. 81–91, 2014.

[15] X. D. Zhang, G. X. Kang, P. Zhang, and H. Zhang, “Game theoretic clustering algorithm for large scale WSN,” Journal of Electronics and Information Technology, vol. 33, no. 10, pp. 2516–2520, 2011.

[16] E. E. Tsiropoulou, G. Mitsis, and S. Papavassiliou, “Interest-aware energy collection & resource management in machine to machine communications,” Ad Hoc Networks, vol. 68, pp. 48–57, 2018.

[17] M. Gupta, N. S. Aulakh, and I. K. Aulakh, “A game theory-based clustering and multi-hop routing scheme in wireless sensor networks for energy minimization,” International Journal of Communication Systems, vol. 35, no. 10, article e5176, 2022.

[18] F. Poloczek and F. Ciucu, “Service-martingales: theory and applications to the delay analysis of random access protocols,” in 34th IEEE Conference on Computer Communications (INFOCOM), pp. 945–953, Kowloon, 2015.

[19] Y. Hu, H. Li, Z. Chang, and Z. Han, “End-to-end backlog and delay bound analysis for multi-hop vehicular ad hoc networks,” IEEE Transactions on Wireless Communications, vol. 16, no. 10, pp. 6808–6821, 2017.

[20] Y. Hu, H. Li, Z. Chang, and Z. Han, “Scheduling strategy for multimedia heterogeneous high-speed train networks,” IEEE Transactions on Vehicular Technology, vol. 66, no. 4, pp. 3265–3279, 2017.
[21] H. L. Sun, X. F. Chi, and L. Qian, “Bandwidth estimation for aggregate traffic under delay QoS constraint based on supermartingale theory,” *Computer Communications*, vol. 130, pp. 1–9, 2018.

[22] S. N. Ethier and T. G. Kurtz, *Markov processes: characterization and convergence*, John Wiley & Sons, Inc., New Jersey, 1986.